VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Diplomová práca

Komprese obrázku pomocí vlnek (wavelets)
Image Compression using Wavelets

2017 Bc. Peter Mikula

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání diplomové práce

Student: **Bc. Peter Mikula**

Studijní program: N2647 Informační a komunikační technologie

Studijní obor: 2612T025 Informatika a výpočetní technika

Téma: Komprese obrázku pomocí vlnek (wavelets)
Image Compression Using Wavelets

Jazyk vypracování: čeština

Zásady pro vypracování:

Cílem práce je implementace vybrané kompresní metody pro obrazy.

1. Přehled kompresních metod.
2. Výběr a návrh vlastní metody komprese obrazu.
3. Implementace a experimenty s navrženou metodou.
4. Porovnání s jinými přístupy.
5. Zhodnocení dosažených výsledků.
6. Závěr.

Seznam doporučené odborné literatury:

[1] David Salomon, Data Compression, Springer 2007.
Články z konference - Conference Data Compression, http://www.cs.brandeis.edu/~dcc/

Formální náležitosti a rozsah diplomové práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí diplomové práce: **prof. RNDr. Václav Snášel, CSc.**

Datum zadání: 01.09.2015

Datum odevzdání: 28.04.2017

doc. Dr. Ing. Eduard Sojka
vedoucí katedry

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

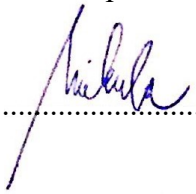
Prehlásenie študenta

Prehlasujem, že som túto diplomovú prácu vypracoval samostatne. Uviedol som všetky literárne pramene a publikácie, z ktorých som čerpal.

Dátum

28. 4. 2017 .....

Podpis



.....

Poďakovanie

Rád by som poďakoval vedúcemu diplomovej práce pánovi prof. RNDr. Václavovi Snášelovi, CSc za odbornú pomoc, konzultáciu, trpezlivosť a podnetné návrhy pri vytváraní tejto práce. Poďakovanie tak tiež patrí mojím blízkym za morálnu a psychickú podporu.

Abstrakt

Cieľom tejto diplomovej práce je implementácia diskrétnej vlnkovej transformácie ako kompresná metóda pre obrazové dáta. V prvej časti je možnosť nahliadnuť do problematiky reprezentácie obrazu ako farebných modelov, či základných grafických formátov. Následne si rozoberieme kompresiu obrazu, najznámejšie kompresné metódy, až sa dostávame k hlavnej téme tejto práce a to je vlnková transformácia. Tu sú popísane pojmy ako vlnka, z ktorých je najznámejšia Haarová alebo Daubechies, dekompozícia, SPIHT kódovanie či realizácia samotnej vlnkovej transformácie. Taktiež sme stručne oboznámení so spôsobom hodnotenia kvality výsledného snímku. V poslednej časti sú porovnané jednotlivé vlnky medzi sebou, ich vplyv na kompresiu a pod. Naše dosiahnuté výsledky sme porovnali s formátom JPEG2000 pomocou softwaru Kakadu. Formát JPEG2000 nebol prekonaný.

Kľúčové slová:

kompresia, diskrétna vlnková transformácia, vlnka, Haar, SPIHT, JPEG2000, PSNR

Abstract

The aim of this diploma thesis is the implementation of discrete wavelet transformation as a compression method for the image data. In the first part, it is possible to look into the problems of image representation as the color models or basic graphic formats. Subsequently, we analyze image compression, the best known compression methods, thus we get to the main theme of this work that is the wavelet transformation. The concepts of wavelet are described here, of which the best known ones are Haar or Daubechies, decomposition, SPIHT coding, or realization of the wavelet transformation itself. Thus we are also briefly acquainted with the way of evaluating the quality of the resulting image. The last part compares the individual ripples between each other, their effect on compression and so on. We have compared our results with JPEG2000 using Kakadu software. The JPEG2000 format has not been overcome.

Keywords:

compression, discrete wavelet transformation, wavelet, Haar, SPIHT, JPEG2000, PSNR

Zoznam použitých symbolov a skratiek

DWT – Discreate Wavelet Transfrom – diskrétna vlnková transformácia

DCT – Discreate Cosine Transform – diskrétna kosínusová transformácia

CWT – Continuous Wavelet Transform – integrálna vlnková transformácia

LIS – List of Insignificant Sets – zoznam nevýznamný množín

LIP – List of Insignificant Pixels – zoznam nevýznamný bodov

LSP – List of Significant Pixels – zoznam významných bodov

HSL – Hue, Saturation, Light – farebný model

HSV – Hue, Saturation, Value – farebný model

HSB – Hue, Saturation, Brightness – farebný model

RGB – Red, Green, Blue – farebný model

CMYK – Cyan, Magenta, Yellow, Key – farebný model

YUV – farebný model používaný v televíznom vysielaní

FIR – Finite Impulse Response – filter s konečnou impulznou odozvou

JPEG – Joint Photographic Experts Group – obrazový formát

JPEG2000 – Joint Photographic Experts Group 2000 – obrazový formát

PNG – Portable Network Graphics – obrazový formát

PCX – PC Paintbrush File Format – obrazový formát

bpp – Bit per Pixel – počet bitov na pixel

PSNR – Peak signal-to-noise ratio – špičkový pomer signálu k šumu

MSE – Mean Squared Error – stredná kvadratická chyba

SPIHT – Set Partitioning in Hierarchical Trees – kódovací algoritmus

EBCOT – Embedded Bitplane Coding with Optimal Truncation – kódovací algoritmus

Obsah

1. Úvod

Predkladaná diplomová práca sa zaoberá problematikou spracovania obrazovej informácie. Konkrétne kompresiou alebo expanziou obrazu pomocou waveletovej transformácie. Digitálne spracovanie obrazu má v dnešnej dobe veľký význam. Prakticky každý signál (obraz) je nutné spracovať, aby mohol byť ďalej využitý. Vstupom môžu byť dáta rôzneho pôvodu, napríklad obrazové dáta z digitálneho fotoaparátu, röntgenové či ultrazvukové snímky vygenerované lekárskym zariadením alebo satelitné snímky z družice atď.

Dnes je úplnou samozrejmosťou, že rôzne dáta si zobrazíme na svojom osobnom počítači, v televízore či mobilnom zariadení. Avšak spôsob, akým sa obrazové dáta zobrazujú na výslednom snímači je otázka určitých algoritmov kompresných metód. V skutočnosti by tieto dáta zaberali neúmernú diskovú kapacitu a pri prenose by zbytočne zaťažovali počítačovú sieť.

V praxi je teda táto kompresná metóda veľmi známa. V nasledujúcej kapitole tejto práce sa bude predstavená problematika farebného priestoru a v ďalšej si predstavíme najznámejšie kompresne metódy. Kapitola 4. nás oboznámi s metódami určovanie kvality obrázka či už pri kompresií, alebo iných transformačných aplikáciách s obrazom.

V nasledujúcej 5. kapitole sa pustíme do rozboru našej hlavnej témy, ktorou je vlnková transformácia, popis najznámejších vlniek, využitie a v neposlednom rade samotná realizácia.

V posledných dvoch kapitolách sa budeme zaoberať implementáciou a použitím príslušných nástrojov a metód k vytvoreniu demonštračného programu k tejto téme a taktiež si vyhodnotíme dosiahnuté výsledky.

2. Reprezentácia obrazu v počítačových systémoch

Obraz chápeme ako vyobrazenie vizuálneho vnímania, obvykle v dvojrozmernej podobe. Dôležitým faktorom u obrazu je fakt, že vo väčšine prípadov reflektuje nejaký objekt, ktorého podoba nie je úplne náhodná, ale existuje v ňom určitá závislosť. Tieto závislosti sú potom uložené aj v obraze a hrajú kľúčovú rolu pri komprimovaní. Kompresia obrazu sa vzťahuje k jeho digitálnej podobe, konkrétne k **rastrovému** typu.

Rastrový obraz je zložený z konečného počtu bodov – pixelov, usporiadaných do matice o konkrétnych rozmeroch, označovaných ako rozlíšenie. Jednotlivé body nesú informáciu o vlastnostiach odpovedajúcej oblasti vyobrazeného objektu (farba, jas).

Ďalším typom je **vektorová** reprezentácia obrazu. Je to popis geometrických objektov a ich vlastností (polohy, farby, prekrývanie atď.). Vďaka tomu, že obsahuje iba parametre objektov, nezávisí na rozlíšení zobrazovacieho zariadenia a obrázky možno ľubovoľne škálovať pri zachovaní kvality. U vektorového formátu sa kompresia veľmi nepoužíva, pretože má malý efekt. Pokiaľ je na súbor vo vektorovom tvare kompresia použitá, tak dochádza ku kompresii celého súboru a kompresia musí byť nestratová [4].

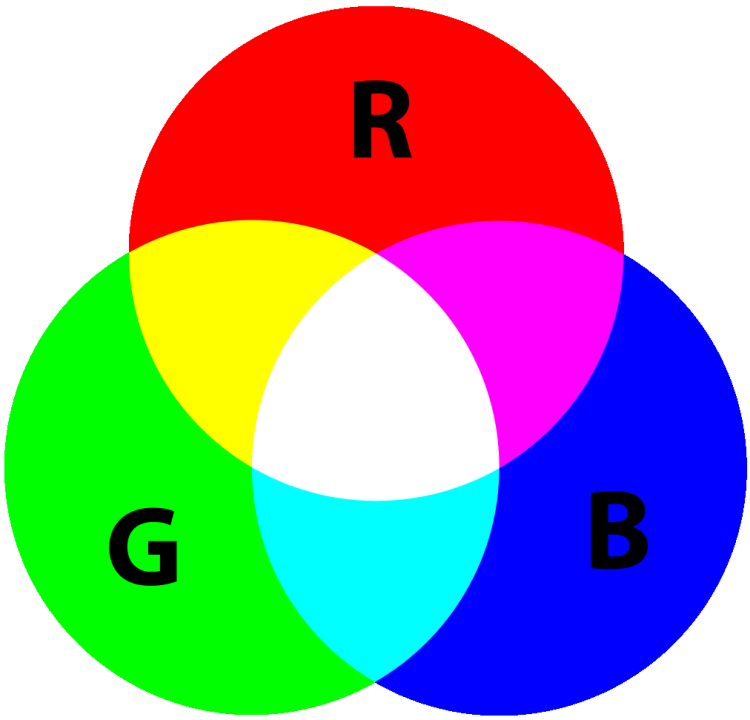
2.1. Farebné modely

Definícií farebných modelov je mnoho. Je možné ho popísať ako abstraktnú matematickú štruktúru popisujúcu farby, ako *n*-tice čísel. Musíme však brať do úvahy, že ide o popis základných farieb a určujú spôsob miešania týchto farieb do farby výslednej. V prírode sa vyskytujú farby, ktoré sú dané zmesou rôznych svetiel vlnových dĺžok, a práve rôzne druhy farebných modelov sa snažia určitú farbu čo najdôveryhodnejšie napodobniť tak, aby bol výsledný vnem pre ľudské oko prirodzený. Farebné modely predstavujú jednu z prvých príležitostí pre kompresiu obrazu. Z vlastností ľudského vnímania možno určiť, ktoré zložky sú menej významne a ktoré, naopak, dôležité. Medzi najznámejšie farebné modely patrí RGB, YUV, YC_BC_R alebo napríklad CMYK. Model CMYK sa zaoberá hlavne farbami typickými pre miešanie maliarskych alebo tlačiarenských farieb a pre účely tejto práce nie je dôležitá. Z ďalších modelov možno zmieniť napr. HLS, HSV alebo HSB [22].

2.1.1. RGB

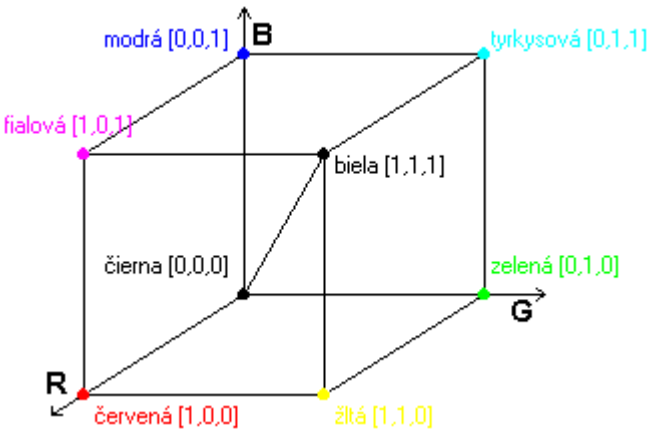
Tento farebný priestor sa v grafických formátoch používa najčastejšie a možno ho označiť za jeden z najzákladnejších [4]. Jeho názov vyplýva z troch základných farebných zložiek, a to červená (R – red), zelená (G – green) a modrá (B – blue).

Základnou vlastnosťou tohto farebného modelu je súčtové alebo aditívne skladanie farieb (obr. 1.1).



Obr. 1.1 Aditívne miešanie farieb v RGB. Zdroj [2].

Princíp spočíva v tom, že si jednotlivé farebné zložky môžeme predstaviť ako žiariace reflektory vyžarujúce svetlo o daných vlnových dĺžkach. Čím viac je farieb skombinovaných (sčítaných, prekrývajúcich sa), tím svetlejší je výsledok [3]. Výsledným sčítaním všetkých farebných zložiek vzniká biela farba. Tento model tiež možno vyjadriť pomocou jednotkovej kocky umiestnenej v osiach r, g, b. Jednotlivé osy súradnicového systému predstavujú jednotlivé farebné zložky farebného priestoru. Vo vrcholoch kocky sa nachádzajú tzv. základné farby. Vo vrchole (0,0,0) sa nachádza čierna farba, naopak v protiľahlom vrchole (1,1,1) farba biela, na telesovej uhlopriečke môžeme vidieť stupne šedej.



Obr. 1.2 Model RGB

2.1.2. YC_BC_R

Tento farebný model vychádza z myšlienky rozdelenia zložiek na jasovú alebo inak odtiene šedej (Y), chrominančnú modrú (C_B) a červenú chrominančnú (C_R). Tento model priamo vychádza z YUV – používa sa predovšetkým v televíznej technike, kde možnosť oddeliť jasovú zložku riešil predovšetkým prechod medzi čiernobielym a farebným vysielaním, kde v prípade čiernobieleho obrazu sa zobrazovala iba jasová zložka Y (luminancia). Ostatné dva farebné signály potom doplnil obraz vo farebný televízny signál.

Ľudský zrak je viac citlivý na jas ako na kontrast farieb. Pri určitej strate informácií v chromatickej zložke je možné docieliť lepšej kvality obrazu pri rovnakej kompresii ako u RGB modelu s tým rozdielom, že by sa chromatické zložky komprimovali viac než jasové.

Obr. 1.3 Obrázok rozložený na jednotlivé zložky Y, C_B, C_R

Samotný prepočet (transformácia) z RGB do YC_BC_R možno vyjadriť nasledovným maticovým usporiadaním:

$$\begin{pmatrix} Y \\ C_B \\ C_R \end{pmatrix} = \begin{pmatrix} 0{,}299 & 0{,}587 & 0{,}114 \\ -0{,}169 & -0{,}331 & 0{,}5 \\ 0{,}5 & -0{,}419 & -0{,}0813 \end{pmatrix} . \begin{pmatrix} R \\ G \\ B \end{pmatrix}.$$

Hodnoty v prvom riadku matice predstavujú konkrétne váhové faktory, ktoré udávajú hodnoty pre jednotlivé farebné zložky. Prvý, modrý farebný chrominančný signál C_B, je vyjadrený rozdielom medzi pôvodnou modrou zložkou a novou jasovou zložkou.

2.2. Najpoužívanejšie obrazové formáty

2.2.1. JPEG

Formát JPEG je jedným z najrozšírenejších formátov pre stratové ukladanie fotografií v počítači. Skratka vychádza z JFIF (JPEG *File Interchange Format*) a skratka JPEG značí názov skupiny, ktorá ju vytvorila (*Join Photographic Experts Group).* Uvedený formát, podporujúci 24 bitovú hĺbku, čo predstavuje 16 777 216 farieb, ukladá obraz pomocou RGB modelu. Typickým príkladom použitia sú snímky uložené na internete. Vďaka jeho veľkej farebnej hĺbke je pre tieto účely viac než dostatočný. Nie je však vhodný na ukladanie textu, ikon alebo perokresby, kde kompresná metóda tohto formátu tvorí rušivé elementy, zvané blokové artefakty. Pre tieto účely sú vhodnejšie formáty PNG alebo GIF. Nesporná výhoda tohto formátu spočíva v jeho kompresii, kde využíva nedokonalosti ľudského oka, teda funguje oveľa lepšie ako bezstratová metóda pri stále veľmi dobrej kvalite výsledného obrazu [5].

2.2.2. JPEG2000

V roku 1997 začal vývoj nového štandardu pre kompresiu obrazu s názvom JPEG2000. Názov samotného formátu značí vtedajší predpokladaný rok uvedenia. Na jeho vývoji spolupracovala už spomínaná skupina *Joint Picture Experts Group*, ale aj skupiny nazvané ISO JPEG *Commitee* a *Digital Image Group.* V roku 2000 sa na svet pozrela prvá špecifikácia a tento formát sa stal otvoreným rovnako ako JPEG. Typické pre súbor JPEG2000 sú prípony *.jp2, *.j2k, *.jpf, *.jpx, *.jpm, *.mj2. Je dôležité spomenúť, že tento štandard nemal za úlohu nahradiť starší JPEG, ale skôr poskytnúť komplementárne funkcie.

Široký záber tohto štandardu umožňuje nasadenie v mnohých oblastiach – od internetu cez medicínske zobrazovanie, až po archiváciu. Každá oblasť vyžaduje špecifické vlastnosti kompresie a formátu dát. Tento obrazový formát je jedným z tých, ktoré používajú pre kompresiu vlnkovú transformáciu, presnejšie jej diskrétnu formu (viď. kapitola 5). Ďalšou zo zaujímavostí je, že dokáže ukladať ako stratovo tak aj bezstratovo za použitia rovnakého kóderu, s čím úzko súvisí rýchlejšia a kvalitnejšia kompresia dát oproti súčasnému JPEG. Rozlíšenie nie je vôbec limitované a na rozdiel od JPEG môže presiahnuť 64000 x 64000 pixelov.

Okrem obrazovej informácie môže obsahovať tiež „metadata“, ktoré tvoria doplňujúce informácie o obrázku, ako napr. názov, dátum a čas vytvorenia, autor, popis, a podobne.

JPEG

JPEG 2000

JPEG

JPEG 2000

Bpp	0.125	0.50	2.00
Image 1 JPEG	24.42	31.17	35.15
Image 1 JPEG 2000	28.12	32.95	37.35
Image 2 JPEG	22.60	28.92	35.99
Image 2 JPEG 2000	24.85	31.13	38.80

Obr.2.2.1 Porovnanie PSNR dvoch obrázkov skomprimovaných pomocou JPEG a JPEG2000 s rôznou bitovou hladinou. Zdroj [6].

Najväčším rozdielom týchto formátov je, že JPEG2000 používa pre svoju kompresiu DWT (kapitola 5.3) a JPEG používa diskrétnu kosínusovú transformáciu (kapitola 3.2.2). Kvôli rozdielnym prístupom nám na obrázkoch vznikli dva rôzne výsledky pri relatívne rovnakej veľkosti súboru. Na obrázku 2.2.1 vidíme, že sa pri vyššej kompresií objavujú rušivé elementy v podobe takzvaných blokových artefaktov, ktoré sú typické pre JPEG. Dôsledkom týchto artefaktov je delenie blokov na dlaždice 8 x 8, s ktorým pracuje DCT. Z uvedeného obrázka vidíme, že pomocou JPEG2000 môžeme pri zachovaní relatívne dobre

pozorovateľného obrazu dosiahnuť väčších kompresných pomerov oproti JPEG. Na záver je nutné podotknúť, že DWT používaná v JPEG2000 má oveľa vyššiu pamäťovú náročnosť a preto sa tento obrazový formát nestal tak rozšíreným ako klasický JPEG.

2.2.3. BMP

Grafický formát BMP [1] (Windows Bitmap) bol po prvýkrát predstavený v roku 1988, následne jeho definíciu firma Microsoft rozšírila a využila ho vo svojom 16-bitovom grafickom operačnom systéme Microsoft Windows 3.0. Veľkou výhodou tohto formátu je, že ho môžeme voľne používať (neexistuje patentová ochrana), ale aj jeho jednoduchosť. Obraz býva ukladaný po jednotlivých pixeloch, čím využíva vlastnosti farebnej hĺbky a počtu bitov na pixel. Navyše môžu pre 8-bitové obrázky používať stupne šedej pre čiernobielu reprodukciu obrazu v 256 odtieňoch šedej. Obrázky uložené vo formáte BMP v prevažnej miere nevyužívajú žiadnu metódu kompresie obrazových dát. To ale neznamená, že tento grafický formát kompresiu nepodporuje. Existujú formáty BMP, ktoré využívajú kompresiu RLE (viď. kapitola 3.1.1). Z dôvodu, že kompresia sa často nevyužíva, sú väčšinou obrázky vo formáte BMP pamäťovo väčšie, ako rovnaké obrázky, ktoré sú ale uložené v inom formáte, ktoré kompresiu práve využívajú (napr. PNG alebo TIFF).

2.2.4. PNG

Obrazový formát PNG (The Portable Network Graphics) sa skladá z hlavičky súboru o dĺžke 8 bytov a zoznamom pomenovaných dátových blokov. Hlavička identifikuje súbor ako PNG, zatiaľ čo bloky poskytujú informácie o špecifických častiach obrazu a rozdeľujú sa na kritické a vedľajšie. Kritické bloky obsahujú informácie potrebné k vykresleniu obrazu (šírku, výšku, farebný typ, bitovú hĺbku, paletu a samotné vzorky), vedľajšie bloky obsahujú pomocné údaje [7].

3. Prehľad kompresných metód

Kompresia je spracovanie dát s cieľom zmenšiť ich objem (jednotka bajt) pri súčasnom zachovaní informácií. Úlohou kompresie (alebo inak aj komprimácie) dát je zmenšiť dátový tok pri ich prenose alebo zmenšiť potrebnú kapacitu pri ich ukladaní. Kompresia v zmysle potlačení nadbytočných informácií sa dá rozdeliť do dvoch základných skupín – **bezstratová** a **stratová.** Rastrová grafika sa vyznačuje vysokou pamäťovou náročnosťou, ktorá rastie kvadraticky s rozlíšením obrazu a preto je žiadúce rastrové dáta komprimovať. Kompresií obrazu v počítačovej grafike je venovaná značná pozornosť.

3.1. Bezstratová kompresia

Umožňuje opätovnú obnovu pôvodných dát – zakódovaný súbor musí obsahovať rovnaké množstvo informácií. Zvyčajne nie je tak účinná ako stratová kompresia dát. Veľkou výhodou je, že komprimovaný súbor možno opačným postupom rekonštruovať do pôvodnej podoby – dekompresia. Používa sa pri prenášaní počítačových dát, výsledkov meraní textu a podobne. Uvedieme si niekoľko najznámejších metód tohto typu.

3.1.1. RLE

Algoritmus Run-Lenght Encoding u nás známy tiež ako prúdové kódovanie. Základným princípom tejto metódy je to, že sa zapíše najskôr počet za sebou opakujúcich sa rovnakých hodnôt a potom samotná hodnota. Reťazec opakujúcich sa znakov sa nazýva prúd. Tento prúd znakov je vždy skomprimovaný do jedného paketu RLE. Príklad vstupných dát kóderu RLE:

„AAAAABBBBCCCDDE“

Výsledný reťazec RLE:

„5A4B3C2D1E“

Výhodou tohto algoritmu je jednoduchosť, ľahká použiteľnosť a vysoká kompresná aj dekompresná rýchlosť. Za nevýhodu môžeme pokladať úzku oblasť dát, na ktorých táto metóda dosahuje dobré kompresné pomery.

V grafických formátoch existujú rôzne varianty RLE kompresie. Buď ako hlavná kompresia dát (napr. PCX) alebo ako pomocná metóda kompresie dát (JPEG).

3.1.2. LZ77

Je založená na princípe nahradzovania duplicitných reťazcov. Kompresná časť algoritmu LZ77 (pomenovaný podľa svojich objaviteľov A. Lempela a J. Ziva a roku vzniku algoritmu 1977) funguje tak, že sa pokúša vyhľadať čo najdlhšie opakujúcu sa postupnosť znakov. Ak takúto opakujúcu sa postupnosť nájde, zapíše na výstup iba odkaz výskytu predchádzajúceho reťazca. Pre zjednodušenie si to môžeme ukázať na príklade. Majme vstupný reťazec:

„ABRAAAKADABRAAA“

Výsledný reťazec by vypadal:

„ABRAAAKAD[9,6]“

Znaky [9,6] je nutné považovať za schematický zapísaný offset udávajúci, že dekóder má z predchádzajúcich deviatich znakov vybrať prvých šesť.

Kompresný algoritmus LZ77 je slovníková kompresná metóda. Podstatou tohto typu je tzv. posuvné okno (sliding window), ktoré je rozdelené na dve časti – **prehliadacie okno** (search buffer, back buffer) a **aktuálne okno** (look-ahead buffer, front buffer). Ich veľkosť je konštantná a vzhľadom k vstupným dátam malá. Dĺžka aktuálneho okna by mala byť vždy menšia alebo rovná dĺžke prehľadajúceho okna. Algoritmus pracuje iba s dátami v týchto oknách, a tým pamäťová náročnosť nerastie s dĺžkou vstupného reťazca.

Algoritmus je inicializovaný tak, že je prehliadacie okno prázdne a do aktuálneho okna sa načíta začiatok vstupu. Následne sa v každom kroku vyhľadá najdlhšie slovo, ktoré začína v prehliadacom okne a je totožné s nejakou predponou (prefixom) slova v aktuálnom okne. Po nájdení je slovo zakódované trojicou **(i, j, z)**, kde **i** je vzdialenosť začiatku slova od začiatku aktuálneho okna (smerom k začiatku okna), **j** je dĺžka slova a **z** je prvý znak nasledujúci po slove. Celé okno je následne posunuté o **j+1** znakov doprava. Toto sa opakuje, až kým aktuálne okno nie je prázdne.

Príklad:

Kompresný reťazec ABRAKADABRA

Veľkosť prehliadaceho okna nech sú 4 znaky a aktuálneho 3 znaky.

Krok	Prehľad. okno	Aktuál. okno	Zvyšok vstupu	Výstup	Posun
Inicializácia	_ _ _ _	ABR	AKADABRA	-	-
1	_ _ _ A	BRA	KADABRA	(0,0,A)	1
2	_ _AB	RAK	ADABRA	(0,0,B)	1
3	_ ABR	AKA	DABRA	(0,0,R)	1
4	BRAK	ADA	BRA	(3,1,K)	2
5	AKAD	ABR	A	(2,1,D)	2
6	ADAB	RA_		(4,1,B)	2
7	DABR	A_ _		(0,0,R)	1
8	ABRA	_ _ _		(3,1,koniec)	2

Tab.3.1 Príklad LZ77 kompresie

Dekompresia skomprimovaného súboru touto metódou je pomerne jednoduchá a rýchla. Vždy, keď dekompresný algoritmus natrafí na ofset udávajúci ukazovateľ a dĺžku reťazca, tak tento reťazec skomprimuje na výstup.

Krok	Vstup	Buffer	Pridanie	Výstup
1	(0,0,A)		A	A
2	(0,0,B)	A	B	AB
3	(0,0,R)	AB	R	ABR
4	(3,1,K)	ABR	AK	ABRAK
5	(2,1,D)	ABRAK	AD	ABRAKAD
6	(4,1,B)	ABRAKAD	AB	ABRAKADAB
7	(0,0,R)	ABRAKADAB	R	ABRAKADABR
8	(3,1,koniec)	ABRAKADABR	A,koniec	ABRAKADABRA

Tab.3.1 Príklad LZ77 spätnej dekompresie

3.1.3. LZW

Kompresný algoritmus Lempel-Ziv-Welch bol vytvorený Abrahamom Lempelom, Jacobom Zivom a Terry Welchom. Bol publikovaný roku 1984 ako vylepšený algoritmus LZ77 a LZ78. Využitie je späté hlavne vo formáte GIF.

Táto metóda má veľmi dobrý kompresný pomer, rýchlu kompresiu i dekompresiu, malé nároky na pamäť a pracuje s celými číslami. Ďalšou výhodou je adaptívna metóda vytvárajúca dynamický substitučný slovník. Za nevýhodu sa dá pokladať rýchly nárast slovníkových kódov odkazujúcich sa na reťazce pôvodného súboru, čo vedie v niektorých prípadoch k zaplneniu pamäte určenú pre slovníkové kódy. Tento nedostatok možno riešiť

rôzne, napríklad „zmrazením“ slovníka pri určitej veľkosti, odstránením dlho nepoužívaných fráz a podobne.

3.1.4. Huffmanovo kódovanie

Huffmanové kódovanie taktiež patrí do skupiny bezstratových kompresií dát. Jeho princíp spočíva v analýze vstupu a početnosti výskytu jednotlivých symbolov, kde najčastejším symbolom je priradený najkratší kód a naopak, tým menej častým kód dlhší. Výstupom je potom sekvencia binárnych čísel. Vyžaduje dva priechody v prvom sa vykoná analýza, potom následne samotné zakódovanie na základe binárneho stromu vytvorenom po prvom prechode.

Uveďme si príklad Huffmanovho kódovania. Formát súboru je nasledovný: prvý údaj nám predstavuje počet znakov abecedy (*n*). V ďalších *n* riadkoch je znak abecedy a jeho pravdepodobnosť výskytu.

8
E 0.05
G 0.05
A 0.1
C 0.1
F 0.1
H 0.1
B 0.2
D 0.3

Tab. 3.3 Vzorový vstup pre huffmanové kódovanie.

Znak	Kód
E	0000
G	0001
H	001
A	010
F	011
C	100
B	101
D	11

Tab. 3.4. Vzorový výstup huffmanovho kódovania.

3.1.5. Aritmetické kódovanie

Toto kódovanie podobne ako Huffmanové je forma entropického kódovania s meniacou sa dĺžkou kódového slova. Prevádza reťazce do iného tvaru tým spôsobom, že prevedie často sa vyskytujúce znaky na menší počet bitov a vzácnejšie znaky potom naopak na väčší počet bitov – rovnako ako u Huffmanovho kódovania, s cieľom zabrať čo najmenej

bitov. Aritmetické kódovanie pracuje na princípe, že zakóduje celé vstupné slovo do jedného čísla alebo zlomku *n*, kde pre *n* platí: $0 \leq n < 1$.

Pre príklad tohto kódovania si vezmeme postupnosť symbolov, ktoré zakódujeme. Majme postupnosť troch znakov X,Y,Z. Každý z týchto znakov nesie rovnakú pravdepodobnosť výskytu $p_X = p_Y = p_Z$. Ak by sme použili jednoduché blokové kódovanie (napr. RLE) tá by zaberalo 2 bity na znak, čo by bolo plytvanie. U aritmetického kódovania reprezentujeme postupnosť ako racionálne číslo na intervale 0 až 2 o základe 3. Každý použitý znak predstavuje jedno číslo 0, 1, 2. Ďalej majme postupnosť znakov „XYYZXY“ a z tejto postupnosti sa stane $0{,}011201_3$. Keďže máme kód v trojkovej sústave, tak ho prevedieme do dvojkovej, aby bola správa vhodná pre prenos. Prevodom nám vznikne $0{,}001011001_2$, čo je predstavuje len 9 bitov, ktorých je o 25% menej ako pri bežnom blokovom kódovaní. Tento algoritmus je preto veľmi vhodný pre dlhé postupnosti, pretože nie sme obmedzovaní počtom miest za desatinou čiarkou. Pri opačnom procese dekódovania vieme, že reťazec mal dĺžku 6 znakov, tak môžeme previesť číslo jednoduchšie z trojkovej sústavy a zaokrúhliť na 6 desatinných miest a odtiaľ máme opäť počiatočnú postupnosť „XYYZXY“.

3.2. Stratová kompresia

Dochádza ku stratám informácie. To znamená, že dáta nemožno obnoviť presne rovnako. Dochádza, alebo môže dochádzať, k istému skresleniu. Stratová kompresia sa využíva hlavne pre kompresiu dát určených ku zmyslovému vnímaniu (obraz, zvuk, video). U týchto dát je dôležitá čo najmenšia veľkosť, a preto malé skreslenie nemusí byť na škodu, pokiaľ však nedochádza k výraznejšiemu zníženiu kvality. Človek si chýbajúce údaje nevšimne, alebo si ich dokáže domyslieť. U obrázkov sú postupne definované farby jednotlivých bodov, pričom sa väčšina farieb často opakuje. Opäť si uvedieme niekoľko najznámejších stratových kompresných metód, ktoré možno kategoriálne zaradiť medzi transformačné kódovanie. Práve k transformácii dát sa používa ortonormálna transformácia. Medzi tieto transformácie možno zaradiť napríklad diskrétnu Fourierovú transformáciu (predovšetkým jej rýchlu formu FFT – Fast Fourier Transform), DCT (diskrétna kosínusová transformácia) alebo práve DWT (diskrétna vlnková transformácia, podrobnejšie sa jej budeme venovať v kapitole 5). Tieto transformácie prevádzajú pôvodné dáta do iných domén (napríklad z časovej oblasti do frekvenčnej), preto je väčšina informácií uložená s menším objemom (veľkosťou), ako pôvodne.

3.2.1. Rýchla Fourierová transformácia

FFT – Fast Fourier transform, je efektívny algoritmus pre výpočet diskrétnej Fourierovej transformácie (DFT) [23] a aj jej inverznej verzie. Práve FFT je veľmi známa z digitálneho spracovania signálu v našom prípade digitalizovaného 2D signálu v podobe obrazovej informácie.

Základným princípom FFT je práve rozdelenie postupnosti vzoriek na dve postupnosti. Jedna obsahuje všetky nepárne vzorky a tá druhá nepárne vzorky diskrétneho signálu DFT. Aby postupnosť mala rovnakú veľkosť, tak musí byť počet všetkých prvkov párny.

3.2.2. Diskrétna kosínusová transformácia

Pre veľmi kvalitné obrazy, s veľa farebnými prechodmi, sa metódy ako RLE a LZW nehodia. Fotorealistické obrazy sa vyznačujú práve tým, že máloktoré dva susedné pixely majú rovnaké farebné odtiene. Práve pre takéto obrázky bola navrhnutá transformačná stratová kompresia, pri ktorej je kompresný pomer riadený požiadavkami užívateľa.

DCT [1], angl. discreate cosine transform, je špeciálnym, novším a sofistikovanejším prípadom diskrétnej Fourierovej transformácie. Obraz v tomto prípade môžeme považovať za dvojrozmerný vzorkovaný signál. Nájdením korelácie blízkymi alebo aj vzdialenejšími obrazovými bodmi (pixelmi) vzniká tzv. medzipixelová redundancia.

Pri transformácií sa vykonáva prevod spracovaného signálu z časovej oblasti do frekvenčnej oblasti. Dôvodom je predpoklad, že práve obrazy reálnych predmetov (napríklad na fotke) neobsahujú veľké množstvo energie vo vyšších frekvenciách, a teda je vhodné zhromaždiť čo najväčšie množstvo dát do malého počtu koeficientov. Dôsledkom tohto kódovania by malo byť zníženie počtu bitov nesúcich viditeľnú informáciu.

4. Metodika k určovaniu kvality

Pri spracovaní obrazu je veľmi dôležité vedieť posúdiť výsledky spracovania. V niektorých prípadoch si možno vystačiť len s vizuálnym posúdením kvality spracovaného obrazu, ale to iba v prípade, kedy rozdiely v obrazoch sú zjavné. Túto subjektívnu metódu pre posúdenie kvality obrazu nie je možné zahrnúť do nejakého numerického algoritmu, preto sa ňou ďalej nebudeme zaoberať. Kvalitu ale možno určiť aj na základe výpočtovej časovej či pamäťovej náročnosti algoritmu. Zavádzajú sa preto metódy pre objektívne posúdenie kvality, ktoré možno využiť vo všetkých oblastiach spracovania obrazu. Rastrová grafika sa vyznačuje vysokou pamäťovou náročnosťou, ktorá rastie kvadraticky s rozlíšením obrazu, a preto je žiaduce rastrové dáta komprimovať. Kompresii obrazu v počítačovej grafike je venovaná značná pozornosť.

4.1. Kompresný pomer

Je to pomer veľkosti pôvodných dát ku veľkosti skomprimovaných dát. Napríklad pri kompresií 20MB súboru do 5MB je pomer 1:4 alebo tiež možno percentuálne vyjadriť ako 25%. Čím menší pomer, tým lepší kompresný výsledok. Kompresný pomer je ovplyvnený voľbou kompresného algoritmu i typom komprimovaných dát.

4.2. Počet bitov na pixel

Môžeme sa tiež stretnúť so zápisom napríklad 12 bpp, čo predstavuje skratka „bits per pixel“ . Je to vyjadrenie koľko bitov zaberá jeden pixel na obraze. Túto hodnotu vypočítame ako podiel počtu bitov na jeden pixel originálneho obrázku a násobok veľkosti pôvodného obrázku vzhľadom k zakódovanému.

4.3. Peak Signal to Noise Ratio (PSNR)

Medzi najjednoduchšie metódy k posudzovaniu kvality obrazu patrí metóda **PSNR -** termín vyjadruje pomer medzi maximálnou možnou energiou signálu a energiou šumu. Tento pomer je často používaný ako metrika kvality medzi pôvodným a komprimovaným obrazom. Kvôli širokému spektru veľa signálov sa udáva v logaritmickej mierke a jednotkou je decibel [dB]. Čím vyššia je PSNR, tým lepšia je kvalita komprimovaného alebo zrekonštruovaného obrazu. Pre 8 bitový signál býva typické rozmedzie hodnôt medzi 20-40 dB. Pre 16 bitové dáta 60-80 dB. Pri výpočte sa využíva tzv. stredná kvadratická chyba – **MSE (Mean Squared Error)**:

$$MSE = \frac{\sum_{M,N}[I_1(m,n) - I_2(m,n)]^2}{M * N}, \tag{4.1}$$

kde M a N je počet riadkov a stĺpcov, teda rozmer a $I_1(m,n)$ a $I_2(m,n)$ sú hodnoty pixelov pôvodného a skúmaného obrazu. Čím nižšia je hodnota MSE, tým nižšia je chyba. Hodnotu PSNR potom možno vypočítať:

$$PSNR = 10\, log_{10}\left(\frac{R^2}{MSE}\right), \tag{4.2}$$

kde R je maximálna hodnota pixelu, tzn. pre 8 bitový signál to je 255.

5. Vlnková transformácia

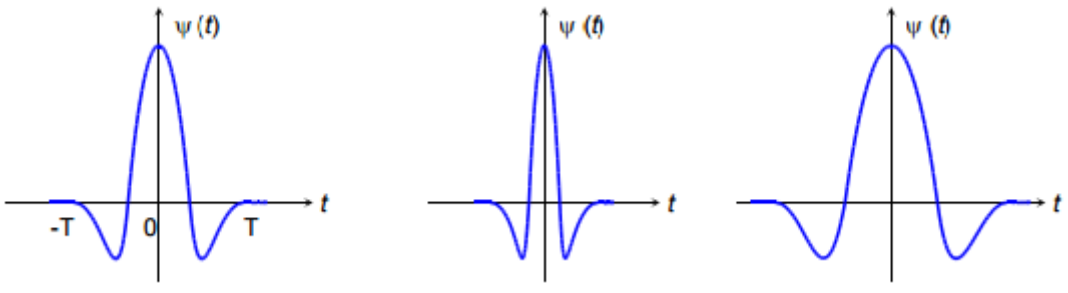
Počiatky vlnkovej transformácie (pôvodne francúzsky ondulette transformation, preložené do angličtiny ako kavalet transform – WT) sú späté s 80. rokmi 20. storočia, kedy boli po prvýkrát použité pre vyhodnocovanie seizmických dát. Ide o istý typ transformácií s rovnakými rysmi, vzájomne sa odlišujúcich podľa tvaru zvolenej bázovej funkcie – vlnky. Odlišujú sa od ostatných transformácií predovšetkým tím, že každá bázová funkcia – vlnka – je podporovaná (tj. má nenulové hodnoty) iba na konečnom časovom intervale, alebo prinajmenšom jej hodnoty mimo tento interval sú zanedbateľne malé. Následkom toho ktorákoľvek hodnota spektra, založená na využití tejto vlnky, je ovplyvnená iba odpovedajúcim úsekom analyzovaného signálu. Teda tiež naopak, vlastnosti, ktoré sú opísané určitou hodnotou spektra, sa môžu vzťahovať k spomínanému konkrétnemu časovému intervalu, čo klasické transformácie neumožňujú. Avšak vlnkové bázové funkcie pokrývajú po častiach celý časový rozsah analyzovaného signálu, takže úplná informácia je zachovaná.

5.1. Bázová funkcia – vlnka

Základným časovo obmedzeným signálom tvoriacim podstatu bázovej funkcie je **materská vlnka ψ**(t) (mother wavelet). Z nej sa odvodzujú vlnky podobného tvaru zmenou mierky a (dilatácia) a posunutím b (translácia). Báza je tvorená množinou funkcií $\psi_{a,b}(t)$ podľa vzťahu

$$\psi_{a,b}(t) = \frac{1}{\sqrt{|a|}}\psi\left(\frac{t-b}{a}\right), a, b \in \mathbb{R}, a \neq 0, \qquad (5.1)$$

kde člen $1/\sqrt{|a|}$ slúži k zachovaniu energie pri zmene mierky. V praxi sa u dilatačných zmien používa väčšinou iba roztiahnutie materskej vlnky (expanzia, $a > 1$).



základná vlnka mierka 0<|a|<1 mierka |a|>1

Obr. 5.1. Zmena mierky vlnky.

5.1.1. Druhy vlniek

Prvá vlnka bola objavená maďarským matematikom menom Alfréd Haar v roku 1909. Avšak pojem vlnka alebo wavelet v tej dobe ešte neexistoval. Až v roku 1984 geofyzik menom **Jean Morlet** a fyzik **Alex Grossman** vymysleli termín vlnka. Dovtedy bola Haarová vlnka jediná ortogonálna vlnka, ktorú ľudia poznali a dokonca veľa vedcov si myslelo, že iná ortogonálna vlnka neexistuje. V roku 1985 matematik **Yves Meyer** skonštruoval druhú ortogonálnu vlnku a nazval ju Meyrova vlnka. Čoraz viac pribúdalo učencov, ktorí sa zaoberali touto oblasťou a v roku 1987 sa uskutočnila prvá medzinárodná konferencia vo Francúzsku. Najznámejšie diskrétne vlnkové transformácie boli formulované belgickou matematičkou **Ingrid Daubechies** v roku 1988. V jej seminárnych prácach odvodzuje rodinu waveletov.

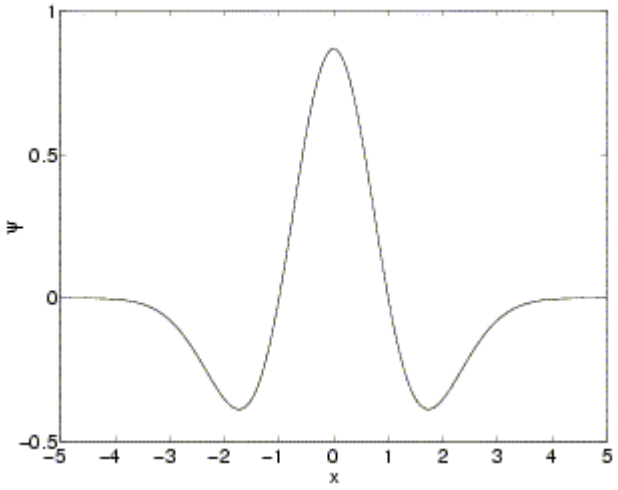
V dnešnej dobe existuje niekoľko stoviek materských vlniek. Vlnky sú kategorizované do rodín podľa podobných vlastnosti, z čoho vychádza aj ich možnosť použitia. U DWT zamerane na kompresiu v obrazoch sa používajú predovšetkým kategórie ortogonálnych a biortogonálnych.

Niektoré typy základných vlniek:

- **Maxikan hat**

Vlnka mexický klobúk alebo tiež Marrová vlnka má tvar druhej derivácie priebehu hustoty pravdepodobnosti Gaussoveho rozdelenia.

Vlastnosti: symetrická, nemá kompaktný nosič, nie je ortogonálna (nemožno použiť pre DWT).

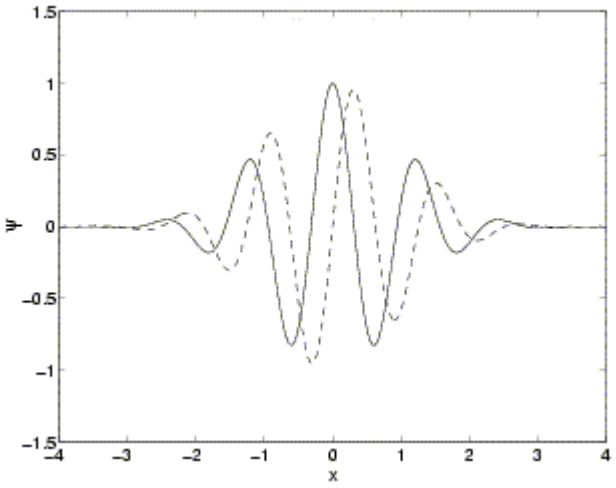


Obr. 5.1.1 Marrová vlnka.

- **Morletová vlnka**

Má tvar komplexnej sínusoidy modulovanej Gaussovským oknom. Jedná sa o kompromis medzi presnosťou polohy a frekvenčným rozložením (FT). Imaginárna časť je vyznačená čiarkovane.

Vlastnosti: symetrická, komplexná, nemá kompaktný nosič, nie je ortogonálna (nemožno použiť pre DWT).

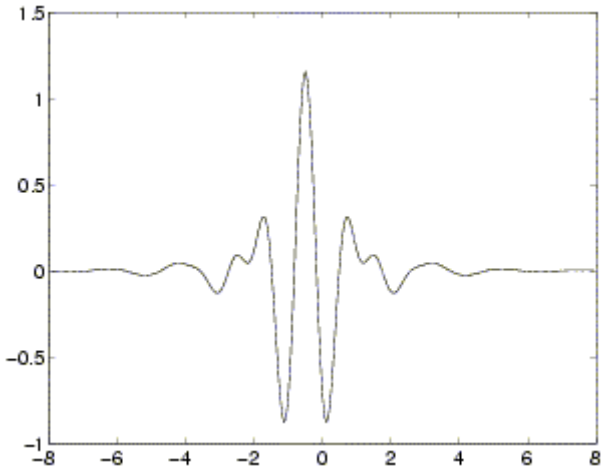


Obr. 5.1.2 Morletová vlnka.

- **Meyerová vlnka**

Je definovaná vo frekvenčnej doméne, nemá explicitný vzorec pre vyjadrenie v čase.

Vlastnosti: symetrická, nemá kompaktný nosič, ortogonálna, je vhodná pre CWT i DWT.

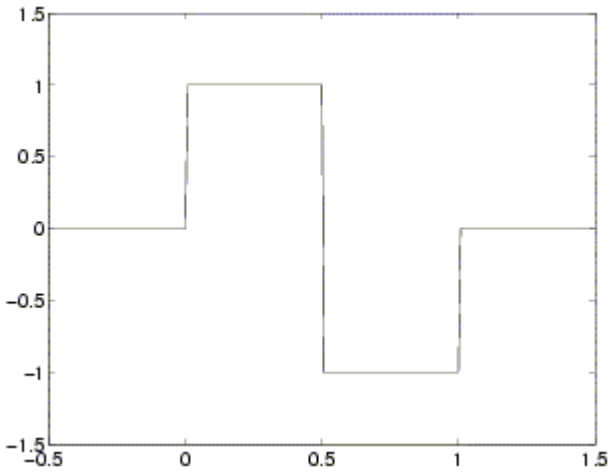


Obr. 5.1.3 Meyerová vlnka.

- **Haarova vlnka**

Predstavuje veľmi jednoduchú vlnku, ktorá ale neumožňuje hladkú rekonštrukciu signálu. Býva často nazývaná Daubechies rádu 1.

Vlastnosti: symetrická, má kompaktný nosič, ortogonálna, je vhodná pre CWT i DWT, je jednoduchšie a efektívnejšie implementovateľná, nespojitá.

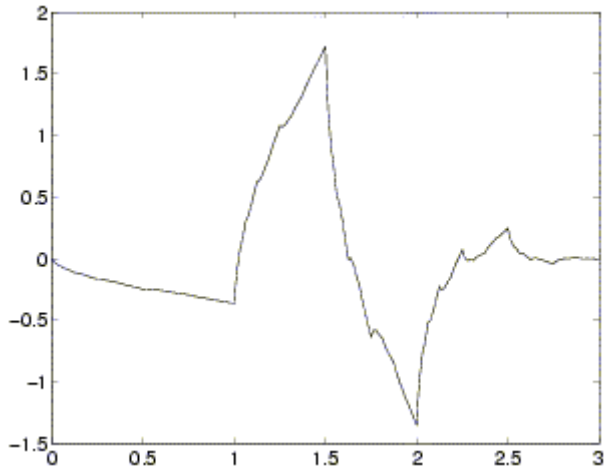


Obr. 5.1.4 Haarová vlnka.

- **Vlnka Daubechies**

Predstavuje skupinu vlniek rôzneho radu N ≥ 1 (N = 1 Haarova vlnka). Nemajú, okrem Haarovej, explicitné vyjadrenie vlnkovej funkcie.

Vlastnosti: asymetrická, má kompaktní nosič dĺžky *2N-1*, ortogonálna, je vhodná pre CWT i DWT.

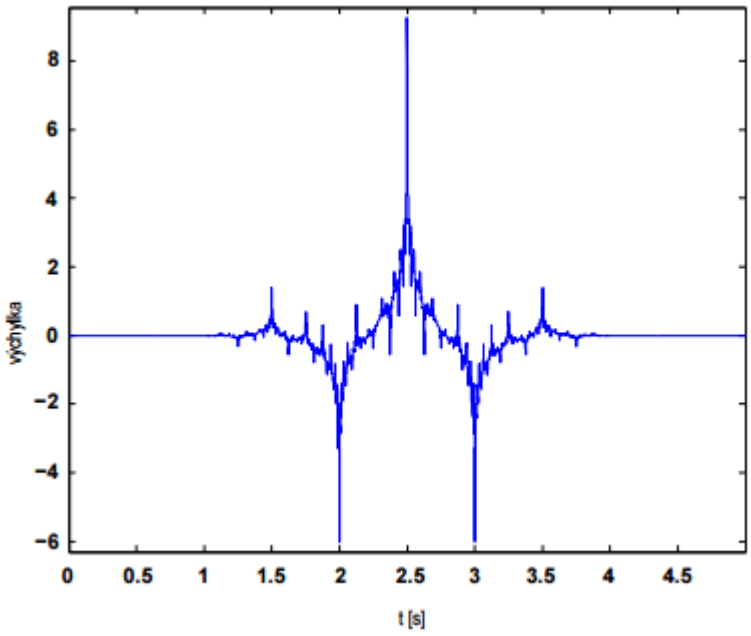


Obr. 5.1.5 Daubechies vlnka.

- **Biortogonálne vlnky**

Rodina biortogonálnych vlniek tak tiež nazývaných Cohen-Daubechies-Feauveau wavelet (v skrat. CDF) [5] bola po prvýkrát odvodená v roku 1992 A. Cohenom, už spomínanou I. Daubechiesovou a J. C. Feauveauom. Jedná sa o rodinu symetrických vlniek, ktoré nemajú explicitné vyjadrenie. Aj keď ich názov nesie meno Daubechies, tak sa nejedná o rovnaké wavelety, pretože si nie sú podobné tvarom ani vlastnosťami.

Ich hlavné využitie je práve v kompresií obrazu a to predovšetkým vo formáte JPEG2000. Ako je známe z predchádzajúcej kapitoly, práve formát JPEG2000 podporuje ako stratovú tak aj bezstratovú kompresiu. Práve pre každý typ kompresie sa v tomto formáte používa iný druh biortogonálnej CDF vlnky. U bezstratovej kompresie sa jedná o biortogonálnu CDF 5/3 vlnku, zvanú tiež ako *LeGall 5/3 wavelet* a naopak u stratovej kompresie CDF 9/7 wavelet. Číselné označenie 5/3 a 9/7 nám udáva dĺžku použitých filtrov typu dolnej priepustnosti.



Obr. 5.1.6 Cohen-Daubechies-Feauveau wavelet 5/3.

Vlastnosti

Existencia kompaktného nosiča – vlnka, pre ktorú existuje kompaktný nosič (*compact support*), má lokalizovanú svoju energiu na konečnom časovom úseku. Táto vlastnosť je výhodná predovšetkým pre rýchlosť výpočtu. Ten možno prevádzať konvuláciou FIR filtra (vlnky) s pôvodným signálom. Tu je známe, že kratšia vlnka – v diskrétnej podobe s menším počtom nenulových koeficientov – predstavuje menej výpočtových operácií.

Symetria vlnky – sa prejavuje pri kompresií obrazu na jeho hranách, kde znižuje nežiaduci efekt nazývaný *ringing*.

5.2. Spojitá vlnková transformácia

Základné vlastnosti sú najlepšie viditeľné z popisu *spojitej vlnkovej transformácie* (CWT –continuous wavelet transform), ktorá je definovaná rovnicou

$$y(a,b) = \frac{1}{\sqrt{a}} \int_{-\infty}^{\infty} x(t)\, \psi * \left(\frac{t-b}{a}\right)\, dt. \qquad (5.2)$$

Jedná sa o časovo frekvenčný rozklad, ktorý je daný korelačným integrálom medzi analyzovaným signálom $x(t)$ a bázovou funciou odvodenou z obecnej komplexnej materskej vlnky $\psi(t)$. Skutočný tvar konkrétnej vlnky $\psi_{a,b}(t)$, ktorá musí mať nulovú strednú hodnotu, závisí na oboch parametroch výslednej funkcie $y(a,b)$. Parameter a označovaný ako mierka (scale) ovláda časovú dilatáciu funkcie (pre a>1 je vlnka natiahnutá a -krát). Parameter ovplyvňuje časový posun funkcie pozdĺž časovej osy, čo umožňuje postupné pokrytie celého časového rozsahu analyzovaného signálu vlnkami určitého konečného trvania. Konštanta $\sqrt{a}$ normalizuje energiu jednotlivých vlniek (viď. kapitola 5.1).

5.3. Diskrétna vlnková transformácia

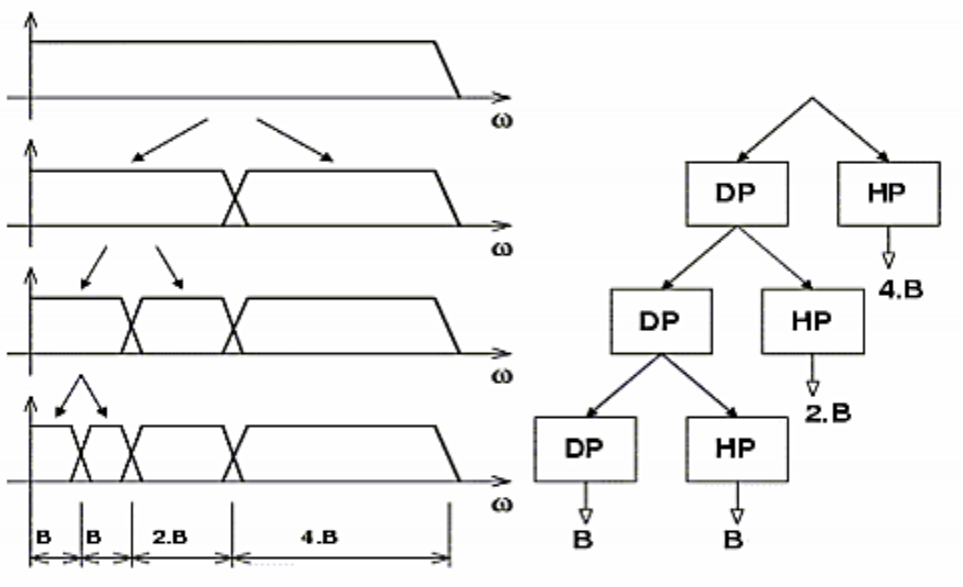
DWT (z ang. Discrete Wavelt Transfrom) sa používa pri spracovaní obrazu, kde dovoľuje redukciu veľkosti dát obrazu na úkor jeho vyšších frekvenčných zložiek bez väčšej straty kvality rozlíšenia. DWT využíva fakt že, ľudské oko vníma v obrázku viac hrán ako okolitú plochu. Hrany (ostré farebné prechody) sú prezentované v spektrálnej oblasti vyššími frekvenciami a v prípade ich odstránenia dochádza iba k rozostrení hrán, ktoré spôsobia len nepatrné zhoršenie kvality obrazu. Týmto môžeme postupne z obrazu odstraňovať tieto frekvencie a tím dosahovať vyššieho stupňa kompresie obrazových dát. Tieto odstránené frekvencie nazývame „detailnými informáciami“, pretože dokresľujú originálny obraz. Táto vlastnosť je zobrazená na obrázkoch nižšie.

Pre číslicovo vyjadriteľnú diskrétnu spektrálnu reprezentáciu je potrebné spojitú transformáciu nejakým spôsobom vzorkovať. Získame ju nahradením parametrov a, b funkciami $a = a_0^m$ a $b = a_0^m kT$, kde $a_0 > 1$, $T > 0$. Koeficienty m, k potom musia byť celočíselné, kde m reprezentuje kmitočtovú mierku a k časovú mierku.

Najčastejším prípadom tohto typu je **dyadická diskrétna vlnková transformácia**, kde parametre $a_0 = 2$ a $m > 0$ potom,

$$y(m,k) = \frac{1}{\sqrt{2^m}} \int_{-\infty}^{\infty} x(t)\psi\,(2^{-m} * t - kT)\,dt. \qquad (5.3)$$

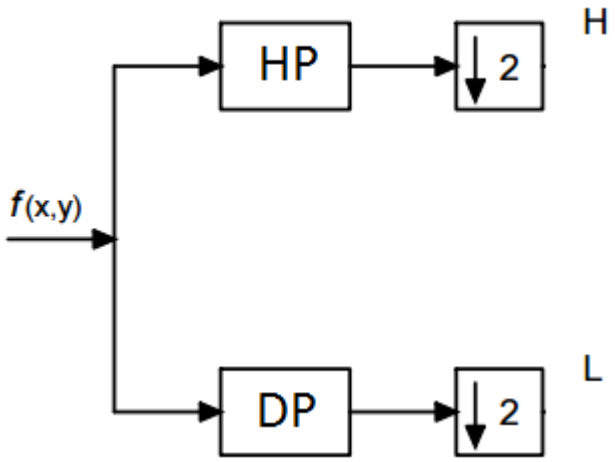
Konštanta T určuje hustotu vzorkovania koeficientov na časovej osi pre jednotlivé kmitočtové úrovne dané indexom m. Konštanta T závisí na šírke pásma B materskej vlnky (ak $T = 1/(2B)$).



Obr.5.3.1 Frekvenčný pohľad na diskrétnu vlnkovú transformáciu

5.4. Realizácia transformácie obrazu

Často býva spracovanie obrazu pomocou DWT realizované istým typom bánk filtrov – **zrkadlové kvadratúrne filtre** (Quadrature Mirror Filter, QMF). Vo všeobecnosti sú banky filtrov označované skupiny filtrov používaných pre rozklad, spracovanie a opätovné zloženie signálu, pričom často dochádza k rozdeleniu vstupného signálu do subpásiem a kanálov. Za základnú banku môžeme považovať dvojkanálovú QMF. Rozdelí vstupný signál pomocou hornej priepustnosti HP a dolnej priepustnosti DP do dvoch frekvenčných pásiem, na nízkofrekvenčný signál obsahujúci **aproximačnú** zložku a vysokofrekvenčný signál s **detailnými** informáciami. Obidva signály sú ďalej podvzorkované s činiteľom 2 – symbol ↓2. Stačí teda zachovať iba párne vzorky a vypustiť nepárne.



Obr.5.4.1 Dekompozícia signálu pomocou bank filtrov

Dekompozícia obrazu u dvojrozmernej vlnkovej transformácie rozdeľuje signál do štyroch časti, ktoré možno nazvať:

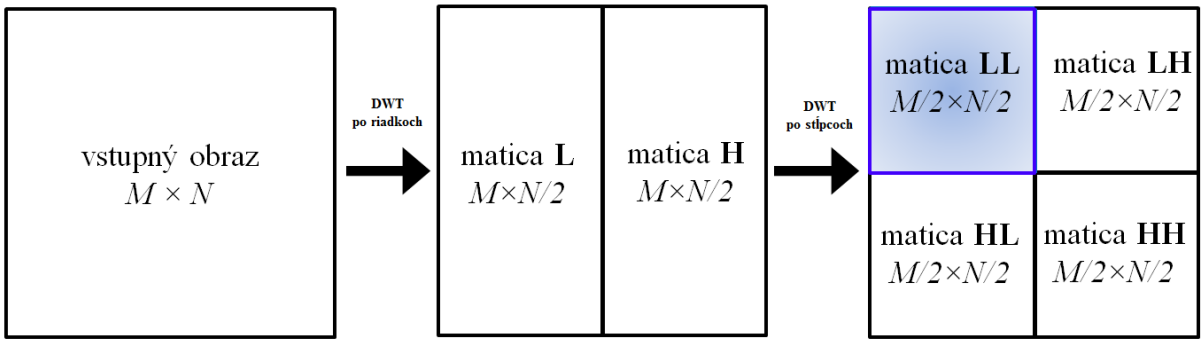
LL – diagonálny aproximačný koeficient, nachádza sa v ľavom hornom rohu **dekompozičného** obrázka, vychádza z dolno-priepustného filtra z oboch smerov (riadky aj stĺpce), zo všetkých 4 častí sa najviac blíži pôvodnému obrázku – preto sa nazýva aproximačný.

Ďalšie tri časti zobrazujú detaily:

HL – vertikálny detailný koeficient,

HH – diagonálny detailný koeficient,

LH – horizontálny detailný koeficient.



Obr.5.4.2 Znázornenie dekompozície.

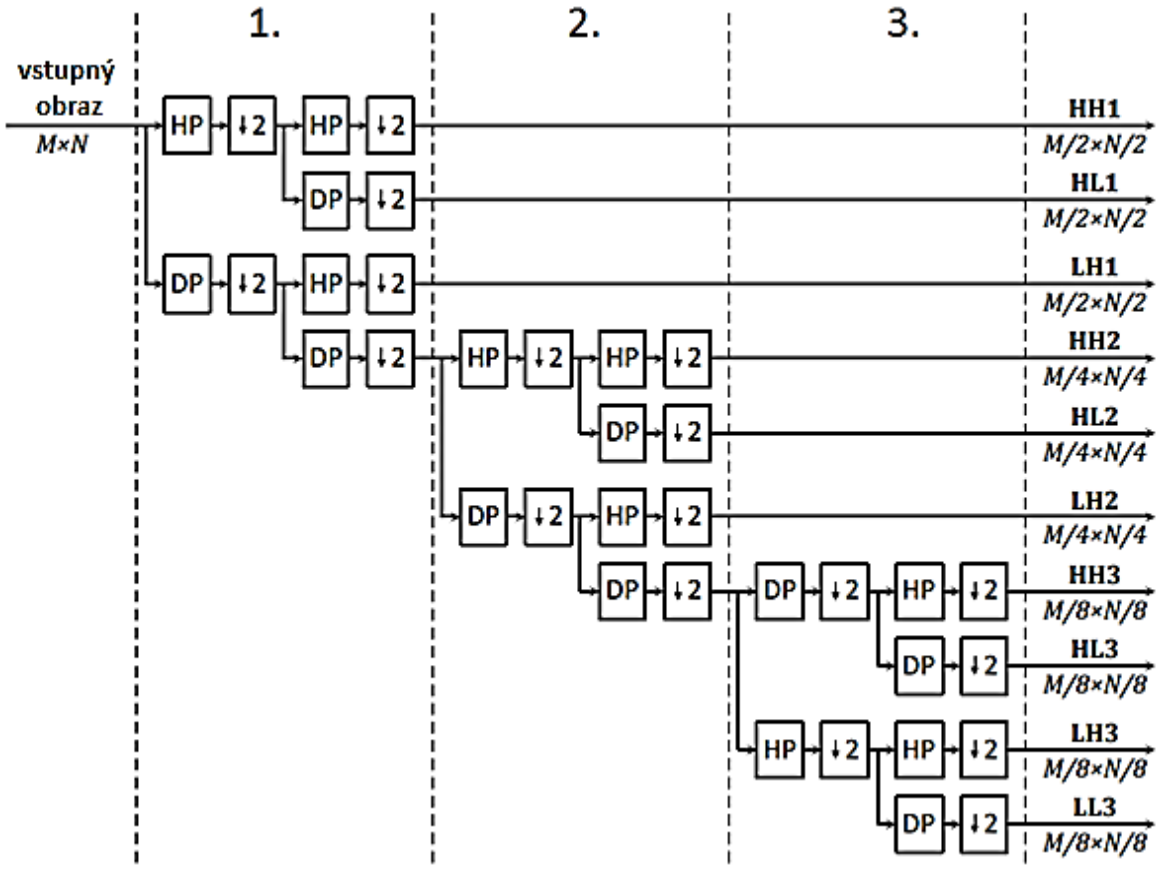
Vstupný obraz je reprezentovaný maticou s rozmermi *M x N*, pričom *M* označuje počet riadkov a *N* počet stĺpcov. Najskôr sa vykoná filtrácia hornej a dolnej priepustnosti s podvzorkovaním, analogicky ako u QMF. Výsledkom je však len rozklad po riadkoch

v dvoch maticiach **L** a **H** o rozmeroch *M x N /2.* Preto sú tieto matice **L** a **H** filtrované rovnakým spôsobom ešte raz. Po vykonaní aj stĺpcového rozkladu dostaneme obraz transformovaný do 4 spomínaných časti (subpásiem). Tieto matice s o rozmermi *M/2 x N/2.* Výsledný obrazec nazývame dekompozičný a jeho príklad je uvedený na obrázku nižšie.

(a) pôvodný obraz *(b) transformovaný obraz*

Obr.5.4.3 Ukážka dekompozície vstupného obrazu 1.stupňa.

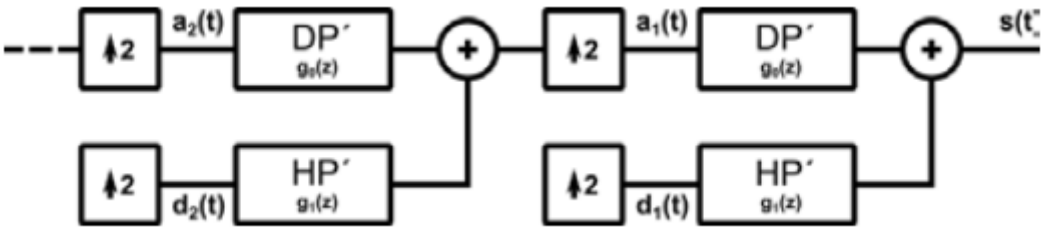
Aproximačná časť **LL** sa potom môže použiť pre druhý stupeň dekompozície. Opakovaním rozkladu získame niekoľko sad detailných koeficientov a pre najvyššiu úroveň dekompozície jednu sadu aproximačných koeficientov – ide teda o akúsi formu rekurzívneho procesu, kde výsledok predchádzajúceho kroku vo forme aproximačných koeficientov je použitý ako zdroj pre ďalší krok.



Obr.5.4.4 Príklad 3-urovňovej 2D vlnkovej transformácie.

5.5. Inverzná waveletová transformácia

Výstupný signál z rozkladovej časti (aproximačný spolu s detailnými koeficientmi) po prvom stupni dekompozície a po prípadnom spracovaní vstupujú do rekonštrukčnej časti. Pretože oba súbory koeficientov majú znížený vzorkovací kmitočet, je nutné najskôr ich vzorkovací kmitočet zvýšiť na pôvodnú hodnotu. Zvýšenie je vykonané pomocou procesu interpolácie s faktorom 2: vkladanie nuly medzi dve vzorky. Oba signály následne prechádzajú filtrom dolnej priepustnosti $g_0(z)$ a hornej priepustnosti $g_1(z)$. Po tejto filtrácií jeden signál obsahuje iba nízke kmitočty a druhý signál obsahuje iba vysoké kmitočty. Výsledný signál z rekonštrukčnej kvadratúry zrkadlovej banky filtrov je súčtom týchto filtrovaných signálov (obrázok nižšie). Ak je výstupný signál z kvadratúrnej zrkadlovej banky filtrov zhodný so vstupným signálom, označuje sa prívlastkom perfektnej rekonštrukcie [8].



Obr.5.5.1 Inverzná WT pre jednorozmerný signál.

5.6. Kódovanie dekompozičného obrazca

Existuje viac prístupov ako na kódovanie dekompozičného obrazca. Jedna veľká skupina metód pracuje so **skalárnou kvantizáciou**. Ide o základnú metódu, ktorá pracuje s jednotlivými koeficientmi ako so samostatnými celkami. Kvantizačný krok sa stanovuje buď globálne alebo výhodnejšie samostatne pre každú úroveň dekompozície zvlášť. Skalárna kvantizácia nahradí pôvodné koeficienty (všeobecne reálne) celočíselnými hodnotami so znamienkom, ktoré vzniknú podelením pôvodného koeficientu tzv. kvantizačným krokom (ktorý sa stanoví napr. pomocou dynamického rozsahu hodnôt v danej úrovni dekompozície). V tomto kroku teda dochádza ku strate informácií.

Po kvantizácií je nutné aplikovať na výsledok bezstratovú kompresiu informácie – najčastejšie sa pracuje s aritmetickým kódovaním [9], prípadne je možné použiť Huffmanové kódovanie, pokiaľ je možných hodnôt medzi koeficientmi viac.

Druhou možnosťou je použiť **vektorovú kvantizáciu.** Pri vektorovej kvantizácií sa snažíme nájsť väzby medzi jednotlivými prvkami a využiť ich v prospech kódovania. Ak zameriame pozornosť na rozloženie vlnkových koeficientov v dekompozičnom obrazci, môžeme zaznamenať istú spojitosť medzi rovnakými skupinami koeficientov v rôznych kvantovacích úrovniach. Každá skupina detailných koeficientov H,V,D sa vzhľadom k najvyššej úrovni dekompozície rozmerovo zmenšuje a koeficienty, ktoré neboli decimáciou vylúčené, naopak naberali vyšších hodnôt. Tento jav je pomerne dobre predikovateľný, pretože každá úroveň je vzhľadom k výške rozmerovo polovičná (dôsledok dyadickej dekompozície). Pokiaľ sa na tento fakt pozrieme čisto z programátorského hľadiska, môžeme pozorovať určitú stromovú štruktúru (každému koeficientu v určitej úrovni odpovedá skupina štyroch koeficientov v úrovni o stupeň vyššie).

Bolo len otázkou času, kedy sa objaví kódovanie, využívajúce vyššie uvedeného postrehu vo svoj prospech. Prvou takouto metódou bolo kódovanie **EZW (Embedded Zerotree Wavelet)** a prišiel s ním v roku 1993 J.M. Shapiro [10]. Ďaleko efektívnejšie

riešenie je bitovo orientované kódovanie **SPIHT (Set Partitioning in Hierarchival Trees)**, s ktorým prišli v roku 1996 A. Said a W.A.Pearlman [11]. Toto kódovanie z veľkej časti vychádza práve z EZW a ďalej ho rozširuje. Metódy sa skladajú z krokov, pri ktorých porovnávame hodnoty koeficientov s určitým prahom, ktorý sa s každým ďalším krokom znižuje. Vzhľadom k využitiu stromových štruktúr a predpokladu zväčšovania hodnôt koeficientov so zvyšujúcou sa úrovňou dekompozície je veľmi pravdepodobné, že veľké množstvo koeficientov ležiacich pod úrovňou prahu sa nám podarí zakódovať niekoľkými málo bitmi. Čo do efektivity kompresie, jedná sa o jedno z najlepších riešení, mimo iné i čo sa týka otázky dostatočnej bitovej variácie (ďalšia kompresia vygenerovaného dátového toku bezstratovou metódou nie je dokonca vôbec nutná).

5.6.1. Set Partitioning in Hierarchival Trees

Ako bolo spomenuté, jeden z algoritmov využívaných pri kompresií obrazu za pomoci DWT je algoritmus nazývaný Set Partitioning in Hierarchival Trees, ktorý sa skrátene označuje ako SPIHT. Rovnako ako EZW tak aj SPIHT pracuje s koeficientmi vzniknutými pyramídovým rozkladom 2D DWT. Algoritmus vychádza z EZW a vo veľa ohľadoch ho aj zdokonaľuje. Algoritmus pracuje s celočíselnými koeficientmi vzniknutými z dekompozície obrazu. Princíp spočíva v tom, že na výstupe kóderu je postupnosť bitov, pričom ich dĺžka určuje ich výslednú kvalitu rekonštruovaného obrazu. Z toho vyplýva, že u SPIHT je možné kompresný postup kedykoľvek zastaviť. Ak tento algoritmus necháme prebehnúť až do konca, tak obrazové dáta na výstupe predstavujú skoro bezstratový obraz iba s malou chybou pri zaokrúhľovaní výpočtu. Z podstaty SPIHT ďalej vyplýva, že bitový tok je progresívne dekódovateľný, to znamená, že obrazová kvalita sa s každým prechodom zlepšuje.

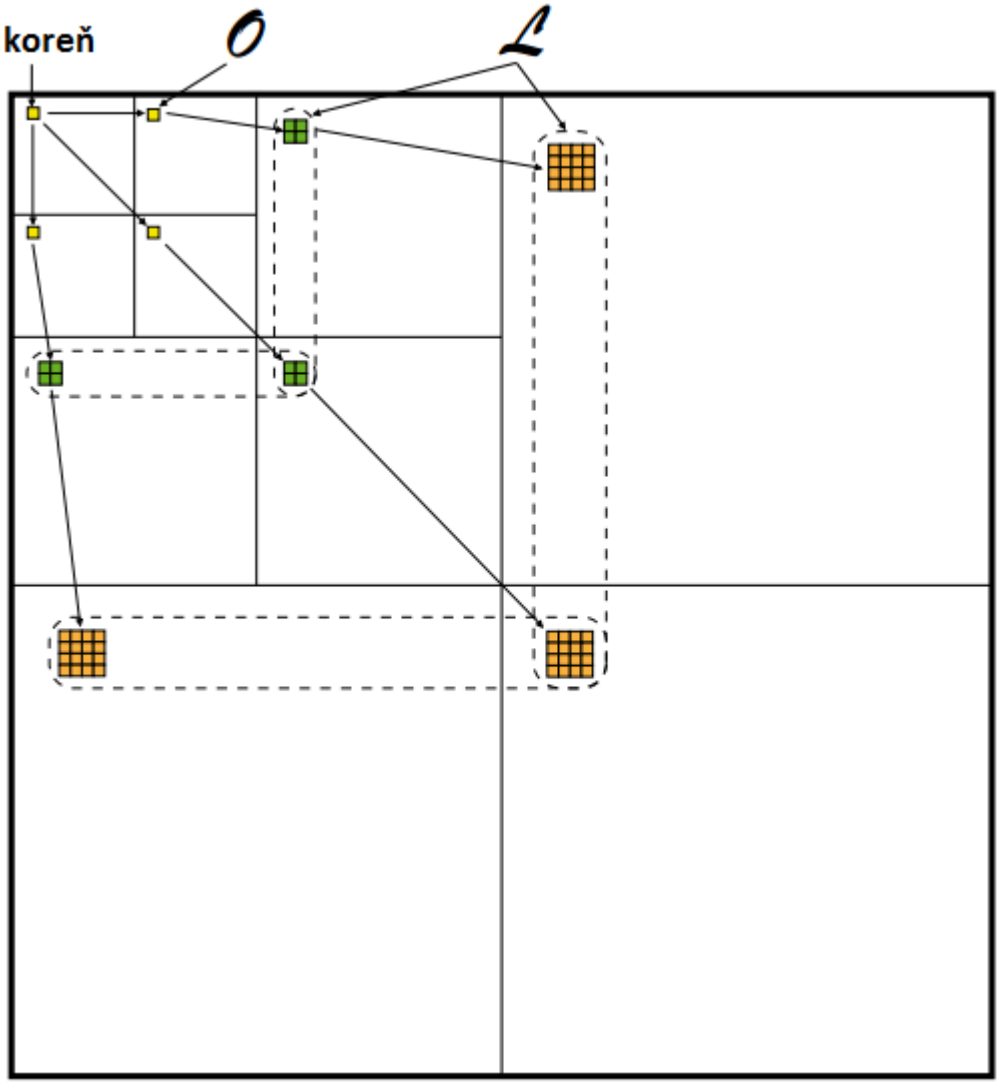
Ak si vezmeme samotný algoritmus, tak ten je založený na dvoch základných princípoch. Prvý je, že budeme uvažovať iba jednu bitovú hladinu. Tá je na počiatku nastavená na bitovú hladinu najvyššieho bitu n najväčšieho koeficientu. Takýmto spôsobom sa vyberú a zoskupia iba také koeficienty, ktoré sú buď väčšie alebo rovné 2^n. Keďže sú očakávané najväčšie koeficienty (koncentrácia energie) v prípadnom najvyššom stupni dekompozície obrazca, tak je tento algoritmus zameraný práve na toto subpásmo.

Druhým princípom je priestorové umiestnenie koeficientov v jednotlivých subpásmach. V každom subpásme sú obsiahnuté koeficienty, ktoré sa vzťahujú k príslušnej oblasti v originálnom obrazce. Toto všetko sa dá znázorniť za pomoci stromovej štruktúry, ako u EZW, z ktorého tento kompresný algoritmus čerpá. Namiesto jednotlivých koeficientov berieme do úvahy celý strom. Na obrázku nižšie môžeme vidieť štruktúru stromu, kde je

vidieť, že korene sa nachádzajú v najvyššom subpásme. Rovnako ako u EZW má každý uzol stromu štyroch priamych potomkov.

Zavádza 4 druhy množín:

- $O(i,j)$: množina súradníc všetkých priamych potomkov (i,j) – offspring,
- $D(i,j)$: množina súradníc všetkých potomkov uzla (i,j) – descendant,
- H:množina všetkých koreňov priestorovo orientovaných stromov (obsahuje indexy najvyššej úrovne dekompozície),
- $L(i,j) = D(i,j) - O(i,j)$: indexy všetkých nepriamych potomkov.



Obr. 5.6.1 Ukážka koreňa stromu a jednotlivých definovaných množín v pyramídovej dekompozícii.

Pre každý prvok sa môže definovať priestorová závislosť jednotlivých prvkov (i,j) v strome. Presnejšie závislosť priamych potomkov prvku (uzlu). Vyjadrenie by sa dalo zhrnúť takto:

$$O(i,j) = \{(2i,2j),(2i,2j+1),(2i+1,2j),(2i+1,2j+1)\}.$$

Tento vzťah platí pre všetky prvky stromu, okrem tých, ktoré ležia v najvyššom alebo najnižšom subpásme.

Rovnako ako je tomu u EZW, sa koeficienty testujú na významnosť $S_n(\mathcal{T})$, kde $\mathcal{T}$ je množina indexov:

$$S_n(T) = \begin{cases} 1 & \max_{(i,j)\epsilon T} |c_{i,j}| \geq 2^n \\ 0 & inak \end{cases} . \qquad (5.7)$$

Keby sme vyjadrili túto funkciu slovne, tak sa jedná o to, či najväčší z absolútnych hodnôt prvkov danej množiny je väčší alebo rovný hodnote 2^n, kde n je aktuálna bitová hladina. V prípade, že je výsledok funkcie $S_n(\mathcal{T}) = 1$, potom množina je v danom priebehu významná, v prípade $S_n(\mathcal{T}) = 0$, potom logicky nevýznamná. S nevýznamnými množinami sa v danom kroku, ktorý určuje hodnota hladiny n, vôbec neuvažuje. Pokiaľ sa táto funkcia aplikuje na jednoprvkové množiny, tak toto rozhodnutie je konečné a na výstup je odoslaná konečná hodnota. V prípade viacprvkových množín, ktoré boli v danom kroku prihliadnuté ako významné, dôjde k rozdeleniu na podmnožiny alebo aj *podstromy* označované ako T_m . Na tieto podstromy je znova aplikovaná funkcia významnosti. Cieľom tohto procesu je nájsť nevýznamné podstromy, ktoré by sa reprezentovali jediným bitom.

Algoritmus SPIHT uchováva informáciu o usporiadaní určitých množín koeficientov. To je realizované pomocou zoznamu nevýznamných množín **LIS**, nevýznamných bodov **LIP** a významných bodov **LSP**. Každý koeficient v zozname je identifikovaný svojimi súradnicami *(i,j)*. Zoznamy LIP a LSP obsahujú jednotlivé koeficienty a zoznam LIS obsahuje položky z množín D a L. Algoritmus SPIHT možno rozdeliť do štyroch základných častí.

Inicializácia je vykonaná jedenkrát. V tomto kroku sú inicializované zoznamy LIS, LIP, LSP a kvantizačný krok.

Zoraďovací priechod zisťuje zmeny vo významnosti koeficientov v zozname LIP a LIS. Vykonáva sa test položiek zoznamu LIP, či sa nestali významnými. Ak áno, tak sú presunuté do LSP. Položky zo zoznamu LIS sú tak tiež testované, či sa nestali významnými. V kladnom prípade sú rozdelené na potomkov, ktorí sú presunutí na koniec zoznamu LIS.

Spresňovací priechod odosiela najvýznamnejší bit, podľa súčasného kvantizačného kroku na výstup pre všetky položky LSP, okrem tých pridaných v poslednom zoraďovacom priechode.

Úprava kvantizačného kroku znižuje kvantizačný krok o 1.

5.7. Využitie waveletov pri spracovaní digitálneho obrazu.

Vlnková transformácia, vďaka svojej vlastnosti extrahovať zo spracovaného signálu rôzne veľké detaily v odlišných smeroch, našla svoje využitie v mnohých oblastiach spracovania a analýzy ako pre jednorozmerné, tak aj pre dvojrozmerné dáta. Jedná sa napríklad o odstránenie šumu [12], kompresií, doostrovanie, zakódovanie informácií [10], rozpoznávanie vzorov [13], detekcia objektov [14], atď. V tejto práci sa zaoberáme hlavne kompresiou dvojrozmerných signálov. O ostaných aplikáciách vlnkovej transformácie sa dá dočítať vo vyššie uvedených zdrojoch.

6. Návrh a implementácia

V tejto kapitole je popísaný návrh a implementácia aplikácie pre kompresiu obrazu za pomoci waveletov. Program je napísaný v jazyku Matlab a k vytvoreniu bol použitý vo verzii R2015b. Na úvod tejto kapitoly si bližšie predstavíme tento program. Možno to chápať aj ako hlavné dôvody pre výber práve tohto programovacieho jazyka. Program Matlab sa používa k riešeniu širokej škály problémov a jeho celé opísanie by bolo veľmi obsiahle. Preto sa v popise zameriame hlavne na tie časti Matlabu, ktoré budeme potrebovať.

6.1. Matlab

MATLAB (MATrix LABoratory = maticové laboratórium) je výkonné interaktívne prostredie pre vedecké výpočty. Spojuje technické výpočty, vizualizáciu dát a programovací jazyk v jednom prostredí. Spoločne s množstvom dostupných modulov tak vytvára ideálny prostriedok pre inžinierov, vedcov, matematikov a učiteľov pri riešení problémov z rôznych odvetví. Aplikáciu vyvíja firma MathWorks a dodáva ju s celou radou rozšírení a toolboxov.

Významnou prednosťou tohto jazyka je jeho jednoduchosť oproti iným známym jazykom. Matlab je oveľa jednoduchší ako napríklad Fortran či C a skrýva obrovský potenciál produktivity a tvorivosti. Za najsilnejšiu stránku je považované mimoriadne rýchle výpočtové jadro s optimálnymi algoritmami. Bol implementovaný na všetkých významných platformách medzi ktoré patrí Windows, Linux, Solaris a Mac. Matlab využíva tzv. **M-súbory** (m-files) – sú to súbory s príponou *.m a obsahujú postupnosť príkazov a povelov systému Matlab.

Otvorená architektúra Matlabu viedla ku vzniku veľa knižníc funkcií, nazývaných toolboxy. Tieto toolboxy rozširujú použitie programu v príslušných vedných a technických odboroch. Pre naše účely nám postačí toolbox zvaný Image Processing Toolbox. Je výkonný, pružný a ľahko ovládateľný nástroj pre spracovanie a analýzu obrazu. Obsahuje rôzne nadstavby pre manipuláciu s farbami, geometriou a štruktúrou obrazu vrátane 2-D transformácií. Na technológii spracovania obrazu sú založené špičkové metódy lekárskej i priemyslovej diagnostiky, analýza dát a automatizácia. Analýza obrazu je nepostrádateľná taktiež v astronómii, geofyzike, ekológii atď. Pre svoju výpočtovú mohutnosť, otvorenosť a štruktúru aplikačných knižníc je Matlab spolu s Image Processing Toolboxom optimálnym nástrojom pre digitálne spracovanie obrazu [15].

6.2. Implementácia

Ak chceme waveletovú transformáciu použiť pre kompresiu digitálneho obrazu, musíme sa na tento obraz pozerať ako na maticu jednotlivých bodov (pixelov). Hodnota každého prvku matice potom udáva jas odpovedajúceho bodu obrazu. V prípade farebného obrazu sa každý jeho bod skladá z troch jasových hodnôt, červená (R - red), zelená (G - green) a modrá (B - blue). Každá z týchto zložiek môže nadobudnúť hodnoty 0 až 255, pričom hodnota 255 je maximálny jas tejto farebnej zložky a hodnota nula znamená, že sa táto farebná zložka na výslednej farbe nepodieľa. Taktiež je možné zvoliť farebný model YC_bC_r (viac v kap. 2.1 Farebné modely). Obraz teda rozdelíme na tri matice, z ktorých každá bude obsahovať jasové koeficienty jednej farebnej zložky.

Ak izolujeme jednotlivé farebné zložky obrazu, získame tri samostatné matice jasových hodnôt. Pre každú z nich vykonáme zvlášť diskrétnu waveletovú transformáciu. Tá je skvelým nástrojom pre kompresiu obrazu, pretože umožňuje obrazový signál rozložiť do rôznych frekvenčných hladín (nízke a vysoké frekvencie) a podľa potreby niektoré frekvencie obrazu eliminovať (viac v kap. 5.4).

Pre vlnkovú kompresiu je možné zvoliť ako vstupný parameter nezápornú malú hodnotu - $\boldsymbol{\varepsilon}$, nazývaná aj ako prahová hodnota (threshold). Všetky prvky transformovanej matice T budú nahradené nulou práve vtedy, ak je absolútna hodnota prvkov v T menšia alebo rovná ako prahová hodnota ε. Matematicky možno zapísať, ak $T = \{a_{ij}\}$ potom

$$B_{ij} = \begin{cases} a_{ij}\, ak\ |a_{ij}| > \varepsilon \\ 0 \quad ak\ |a_{ij}| \leq \varepsilon. \end{cases} \tag{6.1}$$

V závislosti na zvolenej hodnote ε môžeme ovplyvniť typ výslednej kompresie. Ak zvolíme ε = 0, to znamená, že sa nemôžu zmeniť prvky transformovanej matice T, nestratíme pôvodné informácie a môžeme získať pôvodnú snímku späť. Tento typ kompresie môžeme označiť ako bezstratový. Avšak ak zvolíme ε > 0, to znamená, že prvky transformovanej matice T budú modifikované a strácame niektoré detailné informácie. Tento krok znižuje kvalitu obrázka, nemožno získať pôvodný obraz späť. Preto tento typ možno označiť ako stratová kompresia.

Pomerne dôležitá je tiež voľba materskej vlnky. Ako bolo uvedené v obrazovej kompresií, často sa využívajú vlnky Daubechies vo forme biortogonálných filtrov (biortogonalita sa prejavuje najčastejšie rozdielnym počtom prvkov filtrov typu hornej a dolnej priepustnosti – napr. u stratovej JPEG2000 sú použité 9-prvkové filtre typu dolnej

priepustnosti a 7-prvkové filtre typu hornej priepustnosti, ide teda o filtre s konečnou impulznou charakteristikou - FIR). Biortogonálne filtre i cez ich odlišnosť oproti ortogonálnym filtrom splňujú podmienku perfektnej rekonštrukcie. To znamená, že proces dekompozície a následnej rekonštrukcie by mal vrátiť identický výsledok ako pôvodný obraz – bez uvažovania zaokrúhľovanej chyby (ε=0). V našom prípade máme na výber z Haarovej vlnky, vlnky Daubechies radu 4, 8, 16, a biortogonálnej vlnky CDF 9/7. Viac o ich vlastnostiach sa možno dočítať v kapitole 5.1.1 Druhy vlniek.

```
case 1      % ------------------------ HAAR ----------------

    H = [ 1/sqrt(2) 1/sqrt(2) ];       % Haarov hornopriepustny filter
    G = [ -1/sqrt(2) 1/sqrt(2) ];      % Haarov dolnopriepustny filter


case 4     % ------------------------ DAUB4 ----------------

    H = [ -0.1294095226 0.2241438680 0.8365163037 0.4829629131 ];
    G = [ -0.4829629131 0.8365163037 -0.2241438680 -0.1294095226 ];


case 8     % ------------------------ DAUB8 ----------------

    H = [ -0.0105974018 0.0328830117 0.0308413818 -0.1870348117 -0.0279837694 0.6308807679 0.7148465706 0.2303778133 ];
    G = [ -0.2303778133 0.7148465706 -0.6308807679 -0.0279837694 0.1870348117 0.0308413818 -0.0328830117 -0.0105974018 ];


case 16     % ------------------------ DAUB16 ----------------

    H = [ -0.00011747678400228192 0.0006754494059985568 -0.0003917403729959771 -0.00487035299301066...
           0.008746094047015655 0.013981027917015516 -0.04408825393106472 -0.01736930100202211...
           0.128747426620186 0.00047248457399797254 -0.2840155429624281 -0.015829105256023893...
           0.5853546836548691 0.6756307362980128 0.3128715909144659 0.05441584224308161 ];

    G = [ -0.05441584224308161 0.3128715909144659 -0.6756307362980128 0.5853546836548691...
           0.015829105256023893 -0.2840155429624281 -0.00047248457399797254 0.128747426620186...
           0.01736930100202211 -0.04408825393106472 0.013981027917015516 0.008746094047015655...
           0.00487035299301066 -0.0003917403729959771 -0.0006754494059985568 -0.00011747678400228192 ];
```

Obr. 6.2.1 Koeficienty hornej (H) a dolnej (G) priepustnosti u jednotlivých filtrov.

Spomenuté filtre *h(n)* a *g(n)* sú tiež označované ako škálovacie a vlnkové filtre. Škálovací filter *h(n)* prepustí nižšie frekvencie, ktoré obsahujú typicky vyššiu energiu, a reprezentujú približné informácie o signáli. Vlnkový filter *g(n)* prepustí vyššiu frekvenciu obsahujúcu menšiu energiu reprezentujúcu detailné informácie o signáli.

Ďalším krokom implementácie je zvolenie kvantovacieho nástroja. Tu treba mať na pamäti, že pri výbere skalárneho kvantovania je nutné toto doplniť bezstratovou formou kompresie získaných dát, aby bolo výsledné kódovanie efektívne. Takto pracuje napr. formát JPEG2000, kde proces kódovania sa označuje ako EBCOT (Embedded Block Coding with Optimized Truncation).V našom prípade pri zvolenom obrázku v stupňoch šedej, sme zvolili vektorové kvantovanie. U vektorového kvantovania nie je potrebné používať žiadne dodatočné kódovanie, pretože prúd výstupných dát generovaný algoritmom SPIHT je dostatočne variabilný a informačne zhustený. Vektorové kvantovanie ale nie je súčasťou

žiadneho široko rozšíreného otvoreného grafického formátu a pretože SPIHT je metóda patentovaná, svoje uplatnenie nachádza predovšetkým v komerčných, málo používaných formátoch, prípadne pri konkrétnych úlohách (kompresia seizmických dát). Čo sa týka porovnania efektivity vektorového kvantovania oproti skalárnemu, väčšina výsledkov poukazuje práve na vyššiu účinnosť vektorovo orientovaných metód pri kompresii, ale za cenu vyššej výpočtovej náročnosti [16].

Pseudokód algoritmu SPIHT:

--------- *Inicializácia* ---------
$n := [\log_2 (MAX)]$
LSP := prázdny;
LIP := H;
LIS := iba tie prvky z *H,* ktoré majú priamych potomkov, ako prvky typu A;
--------- *Zoraďovací priechod* ---------
for all $(i, j) \in LIP$ **do**
    odošli $S_n (i, j)$;
    if $S_n (i, j) = 1$ **then**
        presuň (i, j) do LSP a odošli znamienko $c_{i,j}$ **;**
    end if
end for
for all $(i, j) \in LIS$ **do**
    if prvek je typu A **then**
        odošli $S_n(D(i, j))$;
        if $S_n(D(i, j)) = 1$ **then**
            for all $(k, l) \in O(i, j)$ **do**
                odošli $S_n(k, l)$;
                if $S_n(k, l) = 1$ **then**
                    pridaj (k, l) do LSP a odošli znamienko $c_{k,l}$;
                end if
                if $S_n(k, l) = 0$ **then**
                    pridaj (k, l) na koniec LIP;
                end if
            end for
            if $L(i, j) \neq$ *prázdny* **then**
                presuň (i, j) na koniec LIS ako prvok typu B;
            else
                odober z LIS prvok (i, j);
            end if
        end if
    end if
    if prvok je typu B **then**

odošli $S_n(L(i, j))$;

if $S_n(L(i, j)) = 1$ **then**

pridaj každý $(k, l) \in O(i, j)$ na koniec LIS ako prvok typu A;

odober (i, j) z LIS;

end if

end if

end for

--------- *Spresňovací priechod* ---------

for all $(i, j) \in LSP$ okrem prvkov pridaných v poslednom zoraďovacom priechode **do**

odošli n-tý najvýznamnejší bit $|c_{i,j}|$;

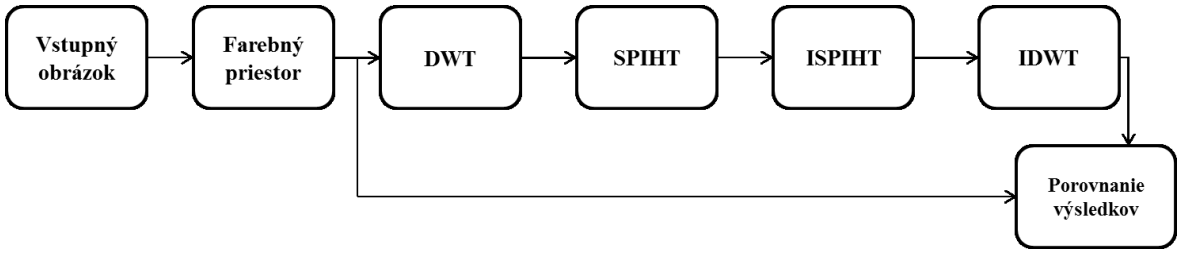
end for

--------- Ďalší krok ---------

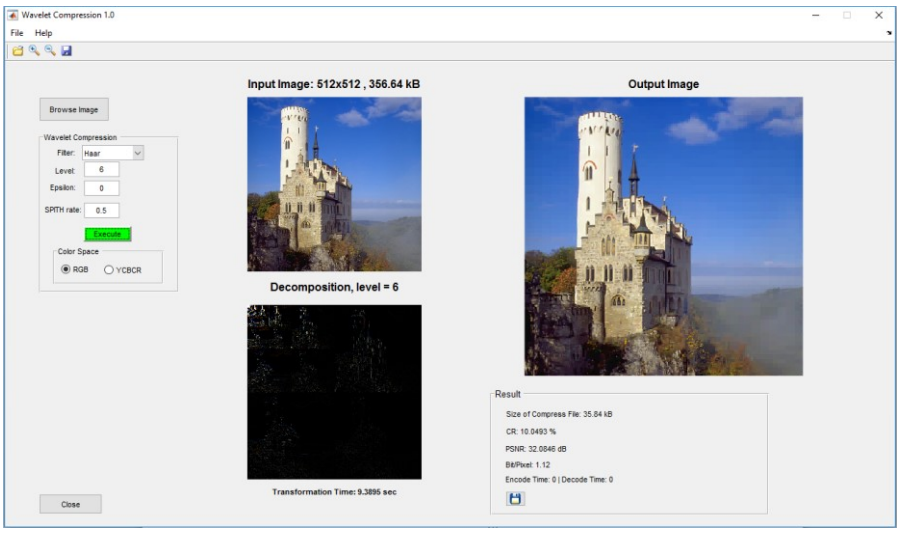
$n := n - 1$;

go to (zoraďovací priechod);

Po vykonaní modifikácií pre všetky farebné zložky a potom opätovné zloženie do jednej matice, získame na výstupe komprimovaný obraz spolu s výsledkami kompresie.



Obr. 6.2.2 Ilustrácia kompresného a dekompresného algoritmu pomocou DWT a SPIHT.



Obr. 6.2.3 Ukážka výsledného užívateľského rozhrania v programovacom jazyku Matlab.

7. Zhodnotenie dosiahnutých výsledkov

V tejto kapitole si vyhodnotíme priamo na príkladoch jednotlivé dosiahnuté výsledky. Následne ich vyhodnotíme pomocou objektívnych metód (PSNR, bpp, kompresný pomer), ktoré sme zhrnuli v kapitole 4.

Súčasťou práce je implementácia kompresného algoritmu s využitím vlniek v prostredí Matlab. Z hľadiska zložitosti sú prispôsobené pre obrazy v stupňoch šedej a o rozmeroch NxN. V našej ukážke ju otestujeme na rôznych obrázkoch a vyhodnotíme ich výslednú kvalitu. V závere kapitoly si test vykonáme aj na farebných modeloch. Na testovanie sme vybrali štandardný a často používaný dataset obrázkov, ako napríklad z [21]. Ako prvý testovací obrázok bol zvolený lena512.bmp o rozmeroch 512x512 v stupňoch šedej – bez uvažovania zaokrúhľovanej chyby (ε=0). Na tomto obrázku si ukážeme podrobnejší rozbor dosiahnutých výsledkov.

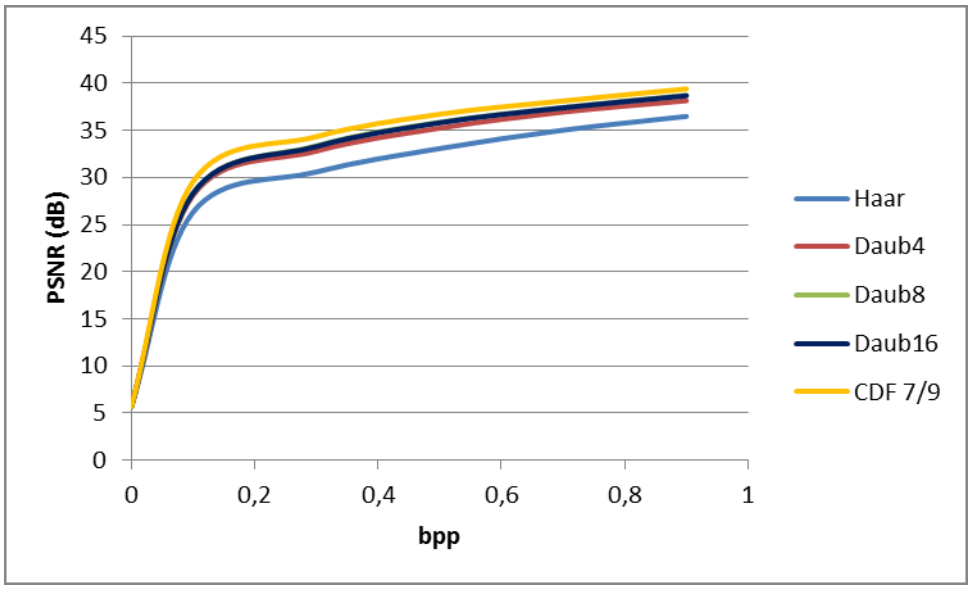
Obr. 7.1 Testovací obrázok Lena,512x512 px.

Ako prvé som sa rozhodol otestovať na danom obrázku rôzne vlnky z našej kolekcie a zistiť ich vplyv pri kompresii. Meranie bolo vykonané na rozklade o hĺbke 6 a za použitia niekoľkých nastavení finálnej kvality (BitRate). Výsledky sú uvedené grafe 7.1, tabuľke 7.1 a

potvrdzujú to, čo sa očakávalo. Filter CDF 9/7 produkuje najlepšie výsledky u kompresie obrazu. Podobne uspokojivé výsledky ukázala aj vlnka Daub8.

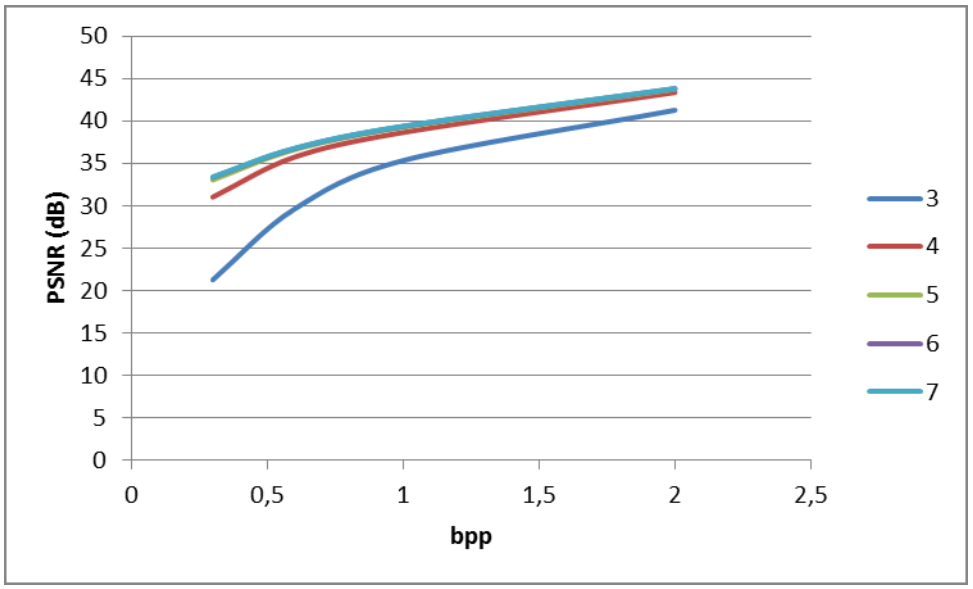
	0	0,1	0,3	0,5	0,7	0,9
Haar	5,6564	26,3128	30,5864	33,0705	35,0202	36,4836
Daub4	5,6564	28,0897	32,8035	35,2365	36,948	38,1451
Daub8	5,6564	28,2926	33,3517	35,8629	37,4342	38,7429
Daub16	5,6564	28,3009	33,2817	35,8067	37,3993	38,6768
CDF 7/9	5,6564	29,5488	34,3338	36,7135	38,1305	39,4071

Tab. 7.1 Testovanie jednotlivých vlniek



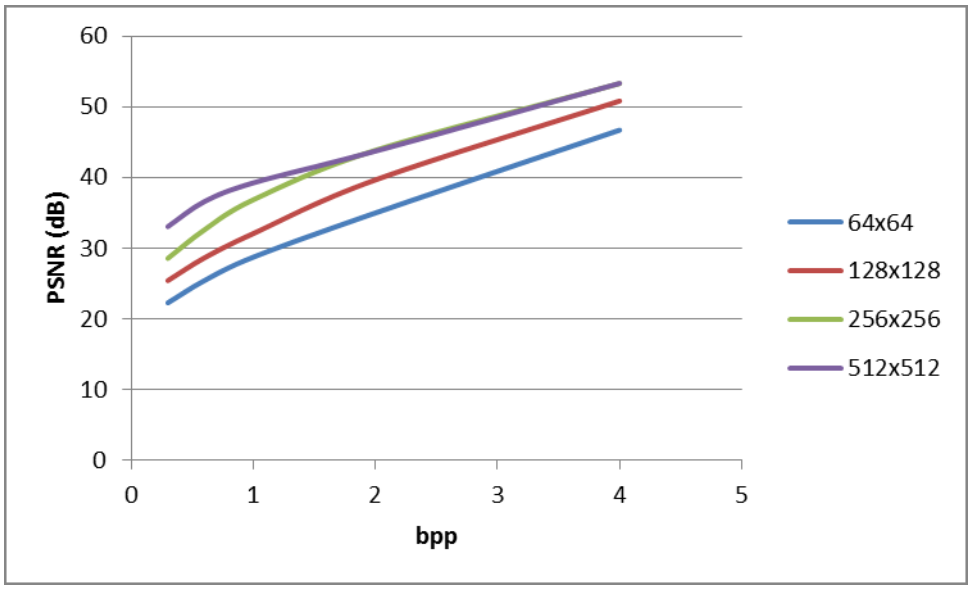
Graf 7.1 Testovanie jednotlivých vlniek.

Na základe predchádzajúceho merania teraz použijeme iba vlnku Daub8. V ďalšom kroku testovania sa snažíme odhaliť vplyv nastavenia úrovne dekompozície na rôznych mierkach bpp. Ako možno vidieť na grafe 7.2, voľba hĺbky rozkladu je dôležitým kritériom, ktoré celú výslednú kvalitu kompresie tak tiež ovplyvňuje. Nastavenie nižšej úrovne znevýhodňuje priestorovú orientáciu stromu. Toto je zaujímavé, pretože 3-levelový rozklad je zvyčajne dostačujúci u iných obrázkoch kompresnej metódy na základe DWT. Táto simulácia nám ukázala, že SPIHT je najúčinnejší pri použití úrovne 5 a vyššie. Rozdiel medzi 3. a 4. úrovňou pri 0.3 bpp je obrovský, a to hlavne z hľadiska subjektívneho hodnotenia. Výsledný obraz pri úrovni 3 je takmer nepoužiteľný, viď. Obr. 7.3.



Graf 7.2 Jednotlivé úrovne dekompozície a ich dopad na výslednú kvalitu.

Tak tiež dôležitým faktorom je to, aký má vplyv na kompresiu rozmer vstupnej snímky. V nasledujúcom teste sa snažíme skomprimovať určité rozmery obrazu Lena s rôznou bpp na úrovni 5. Výsledky sú uvedené v grafe 7.3.



Graf 7.3 Vplyv rozlíšenia vstupnej snímky na výslednú kvalitu.

Z grafu je zrejmé, že vyššie rozlíšenie obrazu vedie k lepším výsledkom výslednej kvality aj kvôli výskytu viacerých detailov v stromovej štruktúre. Tu je žiaduce voliť správnu hĺbku dekompozície pre jednotlivé rozlíšenia na úkor výslednej kvality alebo veľkosti.

Ďalšia simulácia bola zameraná na časovú zložitosť daného algoritmu. Jednoducho povedané sme merali čas v sekundách, ktorý uplynul pri kódovaní a dekódovaní obrazu pri

určitej hĺbke dekompozície a nastavenia kvality. Testovanie bolo vykonané opäť na našom testovacom obrázku Lena 512x512 pomocou Daub8, viď tabuľka 7.2. Z výsledkov je zrejmé, že zložitosť u SPIHT je veľmi nepredvídateľná, pretože výsledky sa líšia a nemožno tvrdiť, že zložitosť presne narastá s vyššou úrovňou dekompozície. To je pravdepodobné vzhľadom na povahu tohto algoritmu, ktorý sa opiera o koncepciu predikovaných stromových štruktúr. Zložitosť je závislá na zdrojovej snímke, hĺbke rozkladu a to nie je ľahko zistiteľné. Kódovacie časy sú znateľne vyššie ako dekódovacie, pretože musí byť aplikovaná kontrola potomkov.

	0,3		0,6		0,9	
level	encode	decode	encode	decode	encode	decode
2	3,5	3,45	6,01	5,42	6,46	5,5
3	0,94	0,43	1,92	0,79	3,96	2,33
4	1,3	0,57	3,39	2,33	7,39	5,88
5	1,51	0,92	4,08	3,16	7,89	6,53
6	1,6	1,02	4,41	3,4	7,81	6,46
7	1,66	1,06	4,36	3,44	7,86	6,42

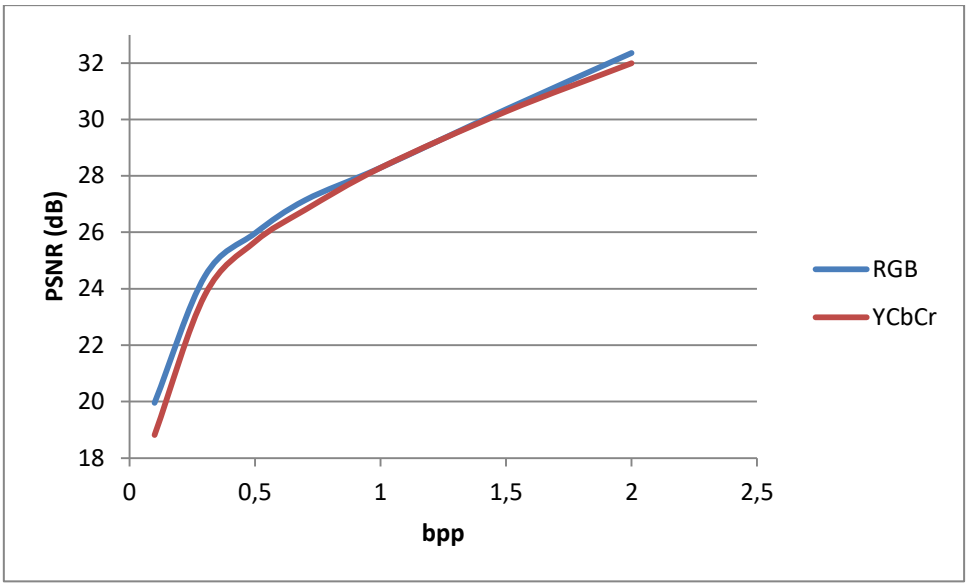
Tab 7.2 Časová zložitosť algoritmu (uplynulý čas v sekundách) na základe rôznych nastavení.

Následne sme sa zamerali na veľkosť samotného zakódovaného bitového toku ktorý obsahuje informáciu o úrovni dekompozície, typ zvolenej vlnky a v poslednom rade samotný bitový tok. Tieto informácie sú potrebné pre spätné dekódovanie. Výsledná tabuľka 7.3 nám ukazuje na to, že zakódovaný bitový tok je menši práve na úkor kvality.

	0,3		0,6		0,9	
level	encode	decode	encode	decode	encode	decode
3	2,4	21	9,54	25,63	22,1	28,6
4	10,7	26,42	26,1	28,6	41,8	30,04
5	14,6	27,24	30,3	28,68	45,8	30,07
6	15,5	27,29	31,3	28,84	46,9	29,9
7	15,8	27,32	31,4	28,85	47	29,81

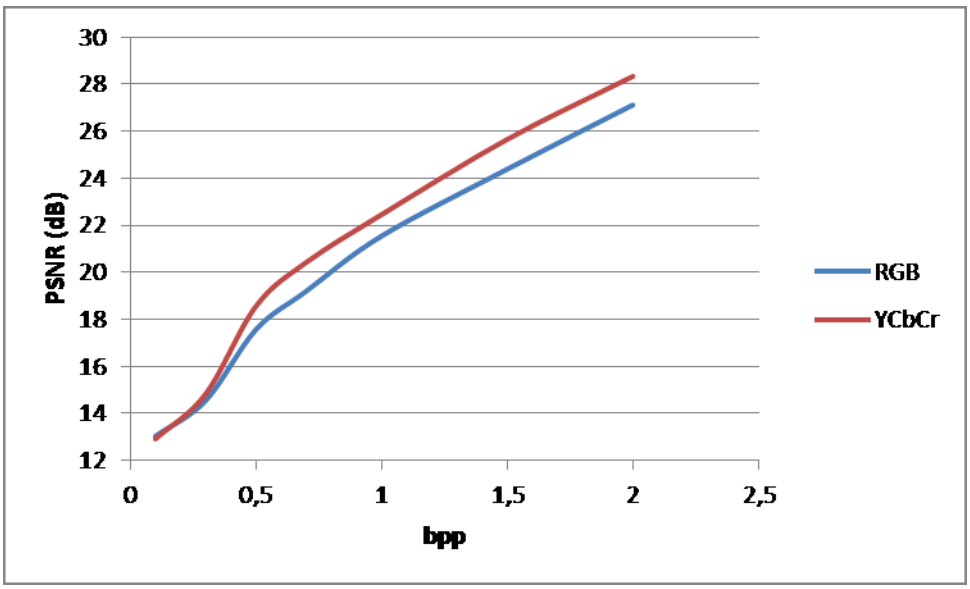
Tab 7.3 Veľkosti zakódovaného a dekódovaného súboru v kB.

V ďalšom testovaní sa pozrieme na porovnanie farebných modelov. Opäť bol použitý algoritmus SPIHT s vlnkou Daub8 a to pre každú farebnú rovinu zvlášť. Avšak tento postup sa javí ako nevýhodný, kvôli problematickej synchronizácií kódovacích postov všetkých rovín. Preto bola navrhnutá modifikácia pre tento algoritmus. Medzi ne možno zaradiť metódu zvanú ako kombinácia jasových zložiek a tiež metódu označovanú ako CSPIHT.



Graf 7.4 Vplyv farebného modelu na kvalitu kompresie obrazu u algoritmu SPIHT a vlnky Daub8, obrázok sunflower.jpg.

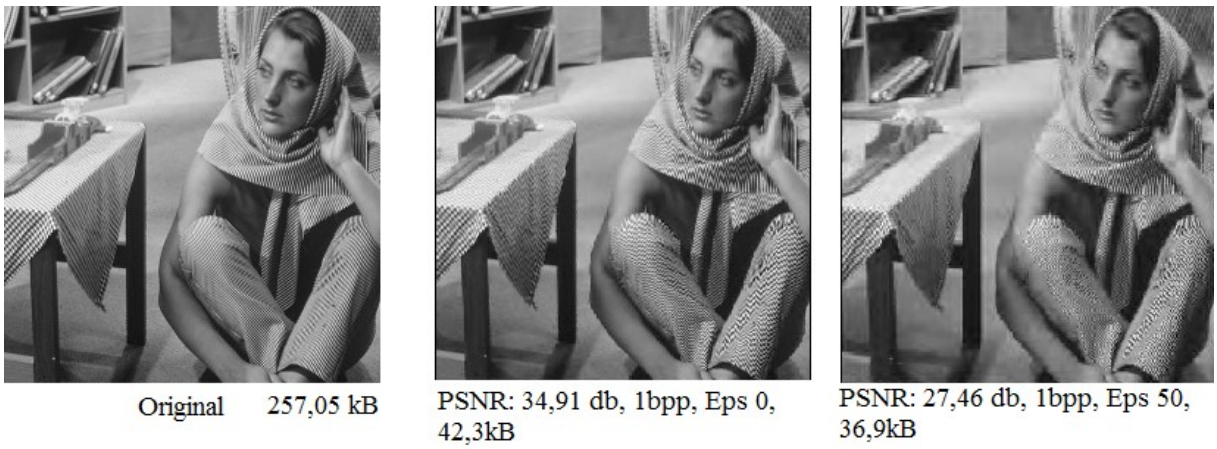
Na grafe 7.4 vidíme, že sa u RGB dosahuje lepšej kvality pri rovnakej bitovej hladine ako u druhého farebného modelu. Pri použití YC_bC_r možno dosiahnuť lepších výsledkov pri vyššom kompresnom pomere. Nie vždy tomu tak je. Na grafe 7.5 vidieť, že u obrázku offset_test.png dosahuje farebný model YC_bC_r výrazne lepších výsledkov.



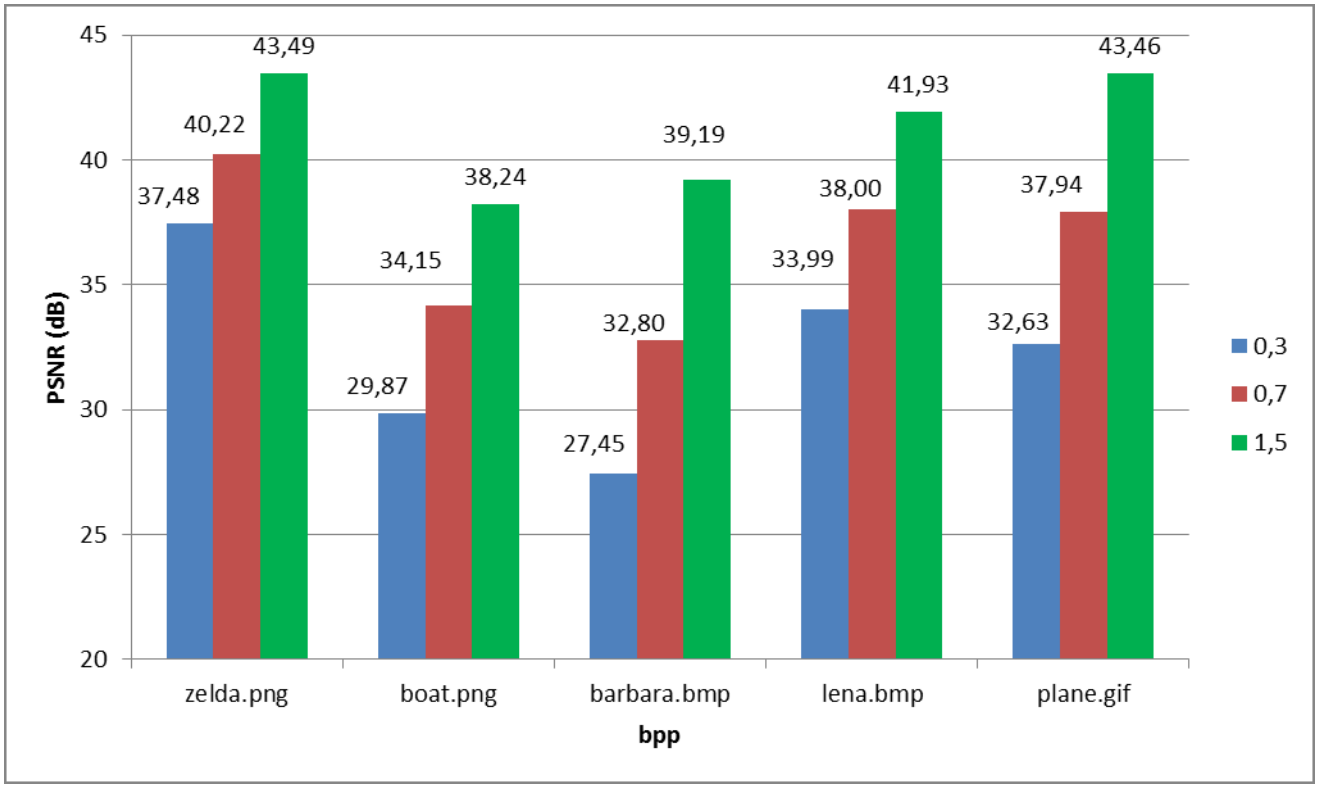
Graf 7.5 Vplyv farebného modelu na kvalitu kompresie obrazu u algoritmu SPIHT a vlnky Daub8, obrázok offset_test.png.

Na záver tejto časti by som chcel predstaviť rôzne príklady výstupu. V ukážke 7.2 a 7.3, prvý obrázok predstavuje vstupný originál s danou veľkosťou ostatne dva znázorňujú výstupný skomprimovaný obraz pri určitých nastaveniach. Graf 7.5 poukazuje na kvalitu

rôznych obrázkov pri určitých bitových hladinách za použitia vlnky CDF 7/9 pri 5. stupňoch dekompozície, s rozlíšením 512×512 px.



Obr.7.2 Výsledné obrázky kompresie pri určitých nastaveniach. Obrázok v strede je nastavený bez zaokrúhľovanej chyby, naopak obrázok na ľavo ju má ale na úkor kvality.



Graf 7.5 Ukážka kvality na rôznych obrázkoch na určitých bitových hladinách. Vstupné obrázky: zelda.png, boat.png, barbara.bmp, lena.bmp, plane.gif; vlnka: CDF 7/9; rozlíšenie:512×512px;level:5.

Obr.7.3 Výsledné obrázky kompresie pri jednotlivých nastaveniach.

Obr. 7.4 Lena512.jpg,CDF 7/9,level 5 pri rôznych bpp.Zľava originál a postupne nasleduje bpp=0,25 PSNR=32,94 dB, bpp=0,125 PSNR=29,50 dB, bpp=0,0625 PSNR=25,53dB.

Obr.7.5 Castle.png, Daub16, level:5. Porovnanie farebnej kompresie so SPIHT kódovaním oproti kompresií bez SPIHT s vloženou zaokrúhľovanou chybou.

8. Porovnanie s iným prístupom

Vlnková transformácia je použitá v niekoľkých formátoch. Ako ich zástupca bol zvolený formát JPEG2000 využívajúci práve vlnkovú transformáciu a pre kódovanie koeficientov používa algoritmus EBCOT. Ako referenčná implementácia formátu JPEG2000 je použitý softwarový balík Kakadu vytvorený *Davidom Taubmanom*. On je tiež autor kódovania EBCOT. V nasledujúcej kapitole nude tento software popísaný.

8.1. Kakadu Software

Kakadu software je vývojársky nástroj pre obrazový formát JPEG2000 a mimo iné obsahuje aplikácie pre konverziu do aj z formátu JPEG2000. Pri písaní softwaru sa brali do úvahy hlavne rôzne druhy aplikácií, pre ktoré mal byť systém určený. Medzi ne patrí kompresia a dekompresia obrazu a videa, prekódovanie medzi podobnými reprezentáciami rovnakého obsahu, interaktívna vykresľovacia aplikácia a klient/server aplikácia. Výhodami systému sú široký základ, z ktorého možno vytvárať aplikácie, vysoká výkonnosť spracovania, nízka pamäťová náročnosť a podpora JPX a JPIP [17].

Jeho nedostatkom je však nízka podpora pre konverziu externých formátov do JPEG2000 a späť. Použitá verzia Kakadu 7.9.0 podporuje kompresiu do JPEG2000 z obrazových formátov PBM, PGM, PPM , RAW , BMP a TIFF a dekompresiu z JPEG 2000 do formátov PGM, PPM, RAW, BMP a TIFF. Nízky počet formátov spôsobuje nutnosť súbor pred kompresiou najskôr skonvertovať do jedného z akceptovaných formátov. To má za následok nie len časové obmedzenie, ktoré môže byť u snímok s vysokým rozlíšením vysoko problematické, ale aj možnú stratu informácií pri konverzii do jedného z vyššie uvedených formátov.

Schopnosti Kakadu demonštrujú programy kdu_*. Rozsah funkčnosti programu je veľmi široký a pohybuje sa v rozmedzí jednoduchých ukážok až po prepracované nástroje. Medzi dva základné a v tejto práci využité programy patrí *kdu_compress* a *kdu_expend.* Prvá uvedená možnosť uskutočňuje práve kompresiu existujúcich obrazových súborov. Výstupom je súbor s jeho zabaleným kódovým prúdom značený príponou **.jp2*.

Program možno spustiť pomocou systémovej konzoly pomocou príkazu:

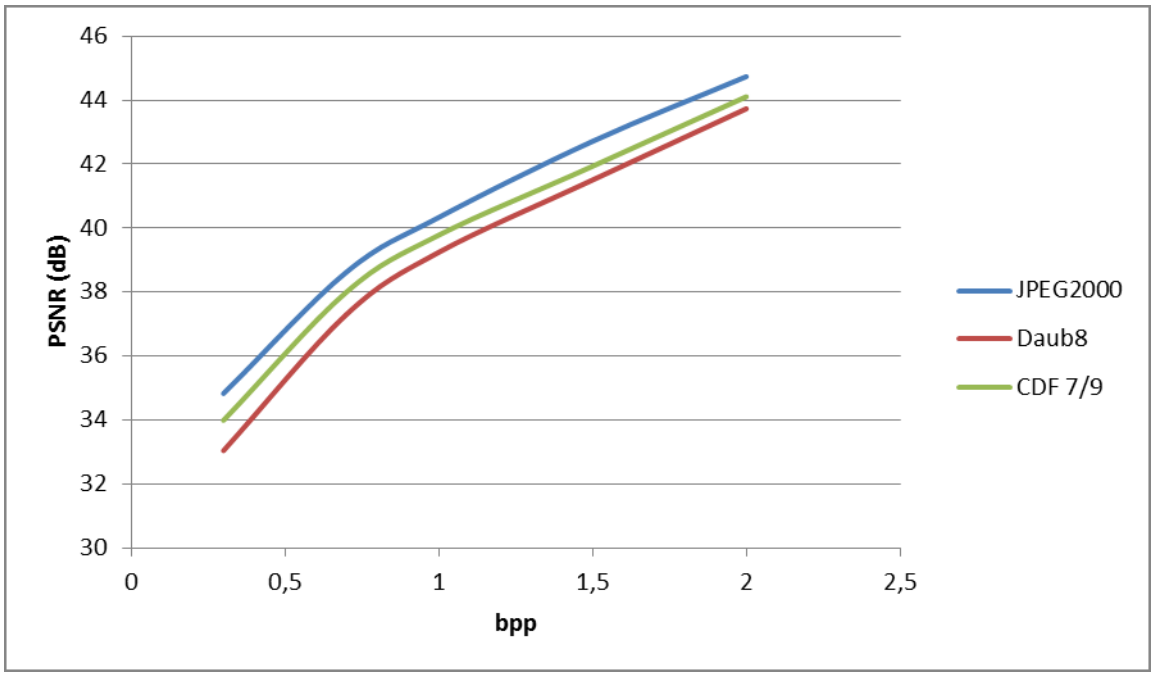
kdu_compress -i lena512.bmp -o filename.jp2 -rate 2,

kde parameter *–i* určuje vstupný súbor, parameter *–o* výstupný súbor a parameter *–rate* slúži k určovaniu výslednej kvality podľa počtu bitov na pixel. Úroveň dekompozície je štandardne predvolená na 5, ale možno ju zmeniť pomocou parametra *–Clevel* [18].

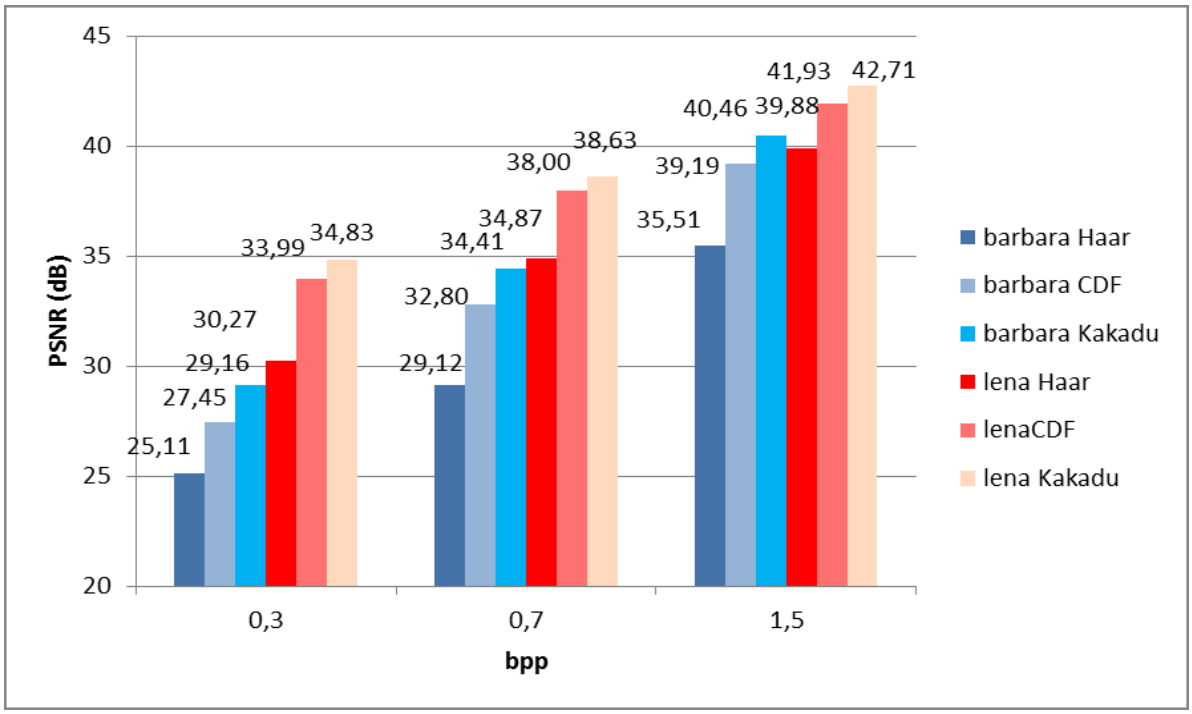
Aplikáciu Kakadu napríklad využíva pri digitalizácii Narodní knihovna ČR pre vytváranie obrázkov vo formáte JPEG2000 [19].

8.2. Porovnanie s JPEG2000

Výsledné porovnanie pre obraz Lena512.bmp, s rozlíšením 512x512 px sú zobrazené v grafe 8.1. Program pre uloženie do formátu JPEG2000 využíva určovanie kvality podľa počtu bitov na pixel.



Graf 8.1 Porovnanie formátu JPEG2000 softwaru Kakadu a implementovaných transformácií s algoritmom SPIHT a kvalitou určenou počtom bitových rovín.



Graf 8.2 Porovnanie kvality na dvoch rôznych obrázkoch lena512 a barbara na určitých bitových hladinách. Na porovnanie bola použitá naša implementácia s vlnkami Haar a CDF 7/9 spolu so SPIHT a Kakadu Software.

Formát JPEG2000 vyšiel z porovnávania ako víťaz, ktorý síce používa vlnkovú transformáciu, ale je u neho použitý algoritmus EBCOT, na rozdiel od implementovaného SPIHT. V rôznych prácach ako napr. u [20] bola preukázaná lepšia efektívnosť algoritmu EBCOT. Software Kakadu je veľmi dobre prepracovaný a optimalizovaný. Taktiež otázka časovej náročnosti u tohto softwaru je neporovnateľne lepšia ako u našej implementácie.

9. Záver

Bol vysvetlený najpoužívanejší postup DWT transformácie dát a ich následné spracovanie. Naznačil som, čo je potrebné pri implementácií jednotlivých algoritmov rešpektovať. Vlnková transformácia má vysoký potenciál pri spracovaní obrazu a to hlavne v oblasti kompresie. Tento potenciál využíva najmä štandard JPEG2000, ktorý vo veľa ohľadoch prekonáva nie len staršie verzie tohto štandardu, JPEG. Experimentálnym testovaním sme si len potvrdili, že za najlepší nástroj, na základe efektivity kompresie, možno považovať práve transformácie založené na DWT.

Ďalej z práce vyplýva, že na účinnosť kompresie má veľký vplyv použitá vlnka (priemerne najlepšie dopadla vlnka CDF 9/7), či nastavenie rôznej úrovne dekompozície, zaokrúhľovacej chyby alebo úroveň bpp. Ďalším skúmaným vplyvom na kvalitu kompresie je farebný model YC_bC_r, ktorý vykazuje lepšiu kvalitu obrazu, za určitých podmienok, pri vyšších kompresných pomeroch. Z môjho pohľadu môžem tvrdiť, že výskum v tejto oblasti zďaleka nemožno považovať za uzavretý – v budúcnosti sa dá očakávať zvýšený záujem o oblasť vektorovej kvantizácie dekompozičného obrazca. Avšak, efektivita kompresie ako taká, nie je rozhodne jediné kritérium, ktorému by sme mali venovať pozornosť. Do hry vstupuje aj veľa iných faktorov, ako napríklad odolnosť kódovania proti chybám v prenosovom reťazci, bezpečnosť pri prenose, možnosť náhodného prístupu do zakódovaného súboru dát a podobne.

Použitá literatúra

[1] SALOMON, David. Data Compression: The Complete Reference. Springer, 2007

[2] RGB color model, In: Wikipedia: the free encyclopedia [online]. Dostupné z: https://en.wikipedia.org/wiki/RGB_color_model

[3] ŽÁRA, Jiří, Bedřich BENEŠ, Jiří SOCHOR a Petr FELKEL. Moderní počítačová grafika: Kompletní průvodce metodami 2D a 3D grafiky. 2. vydání. Brno: Computer Press, 2004.

[4] MURRAY, James D. a William VANRYPER. Encyklopedie grafických formátů.2. vydání. Praha: Computer Press, 1997.

[5] TAUBMAN, David S., MARCELLIN, Michael W. JPEG2000 Image Compression Fundamentals, Standarts and Practice. Kluwer Academic Publishers, 2002.

[6] Comparison between JPEG and JPEG 2000, Dostupné z: http://www.verypdf.com/pdfinfoeditor/jpeg-jpeg-2000-comparison.htm

[7] BMP file format. In: Wikipedia: the free encyclopedia [online]. Dostupné z: http://en.wikipedia.org/wiki/BMP_file_format

[8] VAIDYANATHAN, P. P.: Multirate Systems and Filter Banks. Prentice hall P T R, Englewood Cliffs, New Jersey, 1993.

[9] SAYOOD K.: Introduction to Data Compression, Second Edition, Academic Press / Morgan Kaufmann Publishers, 2000

[10] SHAPIRO, J. Embedded image coding using zerotrees of wavelet coefficients. IEEE Transactions on Signal Processing. December 1993.

[11] SAID A., Pearlman W.A.: A New Fast and Efficient Image Codec Based on Set Partitioning in Hierarchical Trees, IEEE Transactions on Circuits and Systems for Video Technology, vol. 6, 1996.

[12] KOTHER MOHIDEEN, S.; ARUMUGA PERUMAL, S.; MOHAMED SATHIK, M. Image denoising using Discrete Wavelet transform. IJCSNS International Journal of Computer Science and Network Security. January 2008.

[13] ARIVAZHAGAN, S.; GANESAN, L. Texture classificationusing zavelet transform. Pattern Recognition Letetrs. June 2003, pp. 1513-1521.

[14] STARCK, J. L.; MURTAGH, F. Astronomical Image and Data Analysis. Springer-Verlag Berlin Heidelberg, 2002.

[15] MATLAB. In: Wikipedia: the free encyclopedia [online]. Dostupné z: https://en.wikipedia.org/wiki/MATLAB

[16] SWAMINATHAN A., Agarwala G.: Comparative Study Of Image Compression Methods, ENEE631: Digital Image Processing, 2001

[17] Kakadu software. 2014, software. Verzia: 7.9.0, URL http://kakadusoftware.com

[18] Image Class and JPEG2000, URL: http://www.dlxs.org/docs/12/class/image/jpeg2000.html

[19] Národní digitalní knihovna ČR, URL: http://www.ndk.cz/standardy-digitalizace/standardy-pro-obrazova-data

[20] Analysis and Comparison of EZW, SPIHT and EBCOT https://pdfs.semanticscholar.org/c5d0/b56cd7446ca1bcf0121eb106546688e14a53.pdf

[21] DATASET OF STANDARD 512X512 GRAYSCALE TEST IMAGES, Dostupné z: http://decsai.ugr.es/cvg/CG/base.htm

[22] KUEHMI G. Rolf: Color Space and Its Divisions: Color Order from Antiquity to the Present, 2003

[23] Steven W. Smith: The Scientist and Engineer's Guide to Digital Signal Processing, Chapter 8: The Discrete Fourier Transform, Dostupné z: http://www.dspguide.com/ch8.htm

Prílohy

Použité obrázky v testoch

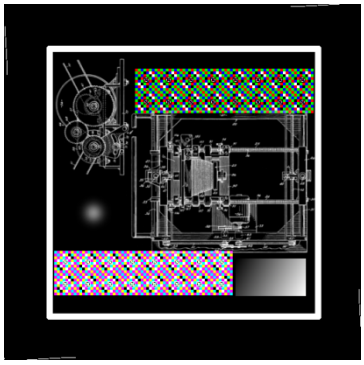
Pr.1 lena512.jpg

Pr.2 barbara.bmp

Pr.3 boat.png

Pr.4 plane.gif

Pr.5 zelda.png



Pr.6 offset_test.png

Pr.7 lena.jpg

Pr.8 sunflower.jpg

Pr.9 castle.png